...SECRETS™ INFO WORLD

“A DOS Treasure Chest!”
—George Yates, President, HAL-PC

# DOS 6 SECRETS™

DOS 6 SECRETS
DOS 6 SECRETS

**Inside!**
Bob’s “Better Than DOS” Shareware Collection—Over 25 Valuable Programs!

Maximize MS-DOS 6 with SECRET Workarounds and Hands-On Solutions

Features Expert DOS 6 Command Reference

All New Coverage of DOS 6 Features! File Compression, Disk Optimization, Backup, Memory Management

ANTI·VI
INTERLINK
EAN ST
DEFRAG

**Multi-Purpose Toolboxes**
ExtraDOS Toolbox
PATRI-SOFT
PocketD v1.64

**File & Directory Management**
ARJ
CMFILER
HOTDIR PLUS
LCD
RELDEL
QEDIT
QFIND
List

**Disk Copying & Formatting**
FORMAT MASTER
DUPE

See Back Cover—
Even More Software!

## by Robert D. Ainsbury

Software Developer & VP of TechnoJock Software, Inc.

**Another Bestselling ...SECRETS™ Book/Disk from IDG**
**From the Publisher of *Windows 3.1 SECRETS™***

IDG BOOKS

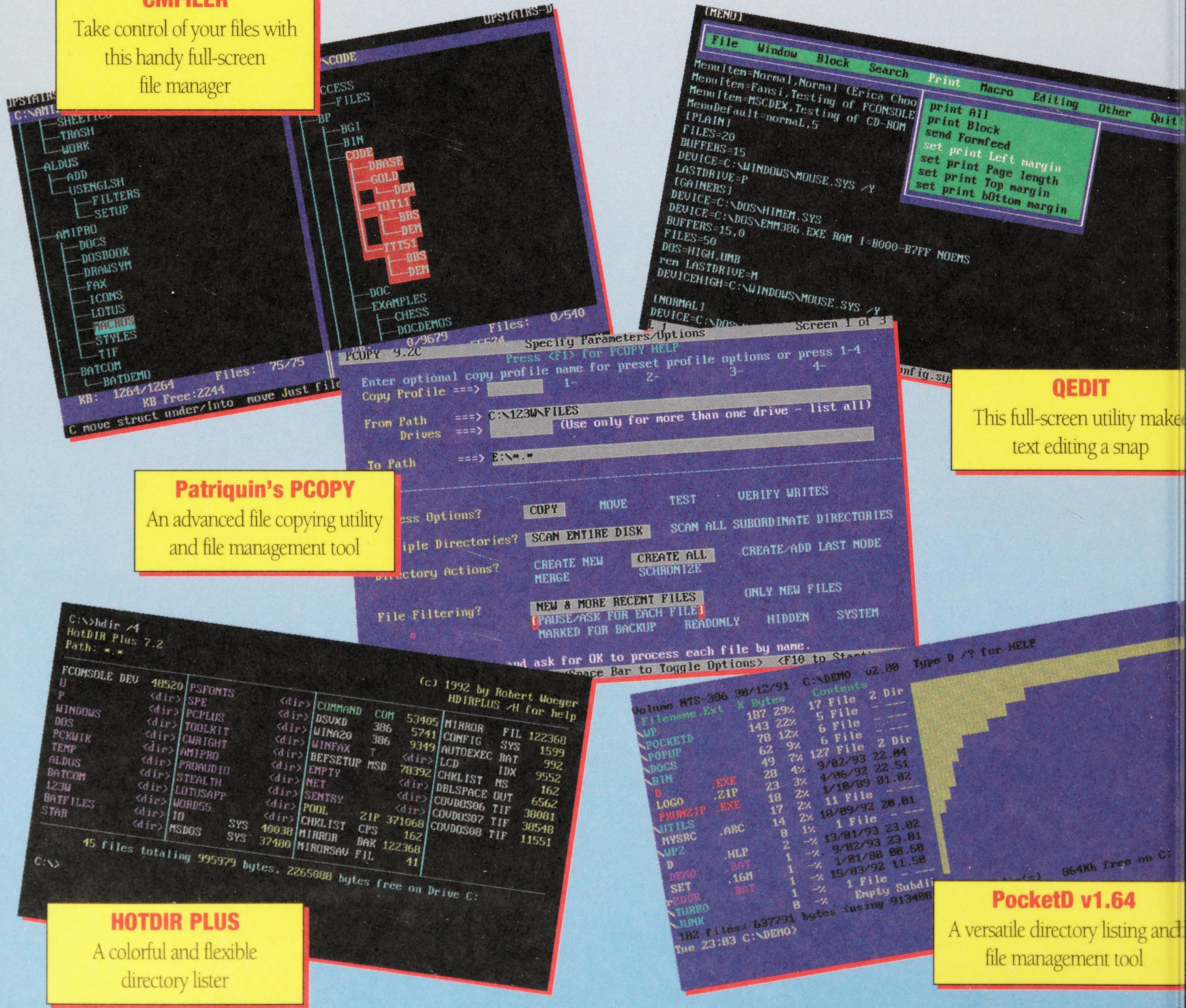
# DOS 6 SECRETS™

## Bob's "Better Than DOS" Shareware Collectio

### Build a better DOS with Bob Ainsbury's exclusive collection of DOS utilit shareware. Featuring the best in disk and file managers, security, batch fil extenders, screen and keyboard extenders, and memory managers.

Over 20 powerful utilities inside, including three multipurpose toolboxes, eight file and directory managers, five hardware enhancers, two disk copying and formatting tools, anti-virus tools, and much more! DOS expert Bob Ainsbury has personally selected these fine programs to make your DOS 6 desktop more powerful than ever.

See inside back flap for complete listing.



**CMFILER**

Take control of your files with
this handy full-screen
file manager

```
UPSTAIRS
C:\AMI
    ─SHEETICO
    ─TRASH
    └WORK
 ─ALDUS
   ├─ADD
   └─USENGLSH
       ├─FILTERS
       └─SETUP
 ─AMIPRO
   ├─DOCS
   ├─DOSBOOK
   ├─DRAWSYM
   ├─FAX
   ├─ICONS
   ├─LOTUS
   ├─MACROS
   ├─STYLES
   └─TIF
 ─BATCOM
   └─BATDEMO
KB:  1264/1264      Files:  75/75
         KB Free:2244
C move struct under/Into  move Just fil
```

```
UPSTAIRS-D
\CODE
CCESS
  ─FILES
─BP
  ├─BGI
  ├─BIN
  ├─CODE
  │  ├─DBASE
  │  ├─GOLD
  │  │  └─DEM
  │  ├─TOT11
  │  │  ├─BBS
  │  │  └─DEM
  │  └─TTT51
  │     ├─BBS
  │     └─DEM
  ├─DOC
  └─EXAMPLES
     ├─CHESS
     └─DOCDEMOS
       0/9679      Files:   0/540
```

```
[MENU]
 File  Window  Block  Search  Print  Macro  Editing  Other  Quit
MenuItem=Normal,Normal (Erica Choo
MenuItem=Fansi,Testing of FCONSOLE
MenuItem=MSCDEX,Testing of CD-ROM
MenuDefault=normal,5
[PLAIN]
FILES=20
BUFFERS=15
DEVICE=C:\WINDOWS\MOUSE.SYS /Y
LASTDRIVE=P
[GAINERS]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM I=B000-B7FF NOEMS
BUFFERS=15,0
FILES=50
DOS=HIGH,UMB
rem LASTDRIVE=M
DEVICEHIGH=C:\WINDOWS\MOUSE.SYS /Y

[NORMAL]
DEVICE=C:\DOS
L 1                          Screen 1 of 3
nfig.sy
```

**QEDIT**

This full-screen utility make
text editing a snap

```
PCOPY  9.2C             Specify Parameters/Options
                    Press <F1> for PCOPY HELP
 Enter optional copy profile name for preset profile options or press 1-4
 Copy Profile ===>            1-          2-          3-          4-

 From Path    ===> C:\123W\FILES
      Drives  ===>              (Use only for more than one drive - list all)

 To Path      ===> E:\*.*

 ess Options?          COPY      MOVE      TEST      VERIFY WRITES
 iple Directories?     SCAN ENTIRE DISK    SCAN ALL SUBORDINATE DIRECTORIES
 ectory Actions?       CREATE NEW    CREATE ALL     CREATE/ADD LAST NODE
                       MERGE         SCHRONIZE
                       NEW & MORE RECENT FILES      ONLY NEW FILES
 File Filtering?      [PAUSE/ASK FOR EACH FILE]   READONLY   HIDDEN   SYSTEM
                       MARKED FOR BACKUP
 nd ask for OK to process each file by name.
    Space Bar to Toggle Options>  <F10 to Start>
```

**Patriquin's PCOPY**

An advanced file copying utility
and file management tool

```
C:\>hdir /4
HotDIR Plus 7.2
Path: *.*
                                              (c) 1992 by Robert Woeger
                                               HDIRPLUS /H for help
FCONSOLE DEV  48520│PSFONTS          <dir>│COMMAND  COM  53405│MIRROR   FIL 122368
U             <dir>│SPE              <dir>│DSVXD    386   5741│CONFIG   SYS   1599
P             <dir>│PCPLUS           <dir>│WINA20   386   9349│AUTOEXEC BAT    992
WINDOWS       <dir>│TOOLKIT          <dir>│WINFAX   T    <dir>│LCD      IDX   9552
DOS           <dir>│CWRIGHT          <dir>│BEFSETUP MSD  78392│CHKLIST  MS     162
PCKWIK        <dir>│AMIPRO           <dir>│EMPTY         <dir>│DBLSPACE OUT   6562
TEMP          <dir>│PROAUDIO         <dir>│NET           <dir>│COVDOS06 TIF  38081
ALDUS         <dir>│STEALTH          <dir>│SENTRY        <dir>│COVDOS07 TIF  38548
BATCOM        <dir>│LOTUSAPP         <dir>│POOL     ZIP 371068│COVDOS08 TIF  11551
123W          <dir>│WORDS5           <dir>│CHKLIST  CPS    162│
BATFILES      <dir>│IO       SYS  40038│MIRROR   BAK 122368│
STAR          <dir>│MSDOS    SYS  37480│MIRORSAV FIL     41│
     45 files totaling 995979 bytes. 2265088 bytes free on Drive C:
C:\>
```

**HOTDIR PLUS**

A colorful and flexible
directory lister

```
Volume NTS-386 30/12/91   C:\DEMO   v2.00  Type D /? for HELP
 Filename.Ext   K Bytes     Contents
\WP              187 29%   17 File  2 Dir
\POCKETD         143 22%    5 File  - ---
\POPUP            78 12%    6 File  - ---
\DOCS             62  9%    6 File  - ---
\BIN              49  7%  127 File  2 Dir
D        .EXE     28  4%   9/02/93 22.04
LOGO     .ZIP     23  3%   4/06/92 22.51
PKUNZIP  .EXE     18  2%   1/10/89 01.02
\UTILS            17  2%   11 File  - ---
MYSRC    .ARC     14  2%  10/09/92 20.01
\WP2               8  1%    1 File  - ---
D        .HLP      2  -%  13/01/93 23.02
DEMO     .BAT      1  -%   9/02/93 23.01
SET      .16M      1  -%   1/01/80 00.50
ZDDR     .BAT      1  -%  15/03/92 11.50
\TURBO             1  -%    1 File  - ---
\JUNK              0  -%    Empty Subdi
 182 files= 637791 bytes (using 913408        864Kb free on C:
Tue 23:03 C:\DEMO>
```

**PocketD v1.64**

A versatile directory listing and
file management tool

# The IDG SECRETS™ Advantage

*DOS 6 SECRETS* is part of the *InfoWorld SECRETS* series of books, brought to you by IDG Books Worldwide, The designers of the *SECRETS* series understand that you appreciate insightful and comprehensive works from computer experts. Authorities in their respective areas, the authors of the *SECRETS* books have been selected for their ability to enrich your daily computing tasks.

The formula for a book in the *SECRETS* series is simple: Give an expert author a forum to pass on his or her expertise to readers. A *SECRETS* author, rather than the publishing company, directs the organization, pace, and treatment of the subject matter. *SECRETS* authors maintain close contact with end users through column feedback, user group participation, and consulting work. The authors' close contact with the needs of computer users gives the *SECRETS* books a strategic advantage over most computer books. Our authors do not distance themselves from the reality of daily computing, but rather, our authors are directly tied to the reader response stream.

We believe that the author has the experience to approach a topic in the most efficient manner, and we know that you, the reader, will benefit from a "one-to-one" relationship, through the text, with the author. The author's voice is always present in a *SECRETS* series book. Some have compared the presentation of a topic in a *SECRETS* book to sitting at a coffee break with the author and having the author's full attention.

And, of course, the author is free to include or recommend useful software, both shareware and proprietary, in a *SECRETS* series book. The software that accompanies a *SECRETS* book is not intended as casual filler. The software is strategically linked to the content, theme, or procedures of the book. We expect that you will receive a real and direct benefit from the included software.

You will find this book comprehensive whether you read it cover to cover, section to section, or simply a topic at a time. As a computer user, you deserve a comprehensive and informative resource of answers that *DOS 6 SECRETS* delivers.

David Solomon
Publisher

INFO WORLD

# DOS 6 SECRETS

by Robert D. Ainsbury

Foreword by [illegible]
President of The Houston Area League of PC Users, Inc.

IDG
BOOKS

IDG Books Worldwide, Inc.
An International Data Group Company

[illegible]

INFO
WORLD

# DOS 6
# SECRETS

by Robert D. Ainsbury

Foreword by George Yates,
President of The Houston Area League of PC Users, Inc.

IDG
BOOKS

IDG Books Worldwide, Inc.
An International Data Group Company

San Mateo, California ✦ Indianapolis, Indiana ✦ Boston, Massachusetts

**DOS 6 SECRETS**

Published by
**IDG Books Worldwide, Inc.**
An International Data Group Company
155 Bovet Road, Suite 310
San Mateo, CA 94402

Text, art, and software compilations copyright © 1993 by IDG Books Worldwide. All rights reserved. No part of this book may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording, or otherwise) without the prior written permission of the publisher.

Library of Congress Catalog Card No.: 93-77602

ISBN 1-878058-70-3

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

Distributed in the United States by IDG Books Worldwide, Inc.

Distributed in Canada by Macmillan of Canada, a Division of Canada Publishing Corporation; by Woodslane Pty. Ltd. in Australia and New Zealand; and by Computer Bookshops in the U.K. and Ireland.

For information on translations and availability in other countries, contact Marc Jeffrey Mikulich, Foreign Rights Manager, at IDG Books Worldwide. Fax: 415-358-1260.

For sales inquiries and special prices for bulk quantities, write to the address above or call IDG Books Worldwide at 415-312-0600.

**Trademarks:** All brand names and product names used in this book are trademarks, registered trademarks, or trade names of their respective holders. IDG Books Worldwide and InfoWorld are not associated with any product or vendor mentioned in this book. InfoWorld is a trademark of InfoWorld Publishing, Inc. ...SECRETS is a trademark of IDG Books Worldwide, Inc.

**Limit of Liability/Disclaimer of Warranty:** The author and publisher of this book have used their best efforts in preparing this book. IDG Books Worldwide, Inc., International Data Group, Inc., InfoWorld Publishing, Inc., and the author make no representation or warranties with respect to the accuracy or completeness of the contents of this book or the material on the disks included with this book, and specifically disclaim any implied warranties or merchantability or fitness for any particular purpose, and shall in no event be liable for any loss of profit or any other commercial damage, including but not limited to special, incidental, consequential, or other damages. See also the next to last page of this book for details regarding the included disks.

# Dedication

To Mum and Dad with love

# Acknowledgments

My fabulous wife, Erica, for supporting me throughout the six-month ordeal and for keeping the business afloat while I was running in protected mode. George Bean, for all the moral support and for the painstaking research on various hardware platforms. Oh, and Uncle Charles, for the secrets that got away.

The folks at Microsoft, especially Andy Thomas and Ben Slivka. True professionals.

The fantastic IDG team. I submitted a donut and they transformed it into a *gateaux!* The marshalls: Mary Bednarek, Terrie Lynn Solomon, and David Solomon. The wordsmiths: Sandy Blackthorn, Diane Steele, Becky Whitney, Chuck Hutchinson, Julie King, and Tricia Reynolds. The Shareware sultans: Laurie Smith and Darrin Strain. The production posse: Cindy Phipps, Beth Baker, and Mary Briedenbach. The cover girl: Polly Papsadore. And Linda Slovick, for all the technical support, corrections, and suggestions.

It was a pleasure working with y'all.

(The publisher would like to give special thanks to Patrick J. McGovern, without whom this book would not have been possible.)

# About IDG Books Worldwide

Welcome to the world of IDG Books Worldwide.

IDG Books Worldwide, Inc., is a division of International Data Group (IDG), the world's largest publisher of computer-related information and the leading global provider of information services on information technology. IDG publishes over 190 computer publications in 60 countries. Thirty million people read one or more IDG publications each month.

If you use personal computers, IDG Books is committed to publishing quality books that meet your needs. We rely on our extensive network of publications, including such leading periodicals as *Macworld*, *InfoWorld*, *PC World*, *Computerworld*, *Publish*, *Network World*, and *SunWorld*, to help us make informed and timely decisions in creating useful computer books that meet your needs.

Every IDG book strives to bring extra value and skill-building instruction to the reader. Our books are written by experts, with the backing of IDG periodicals, and with careful thought devoted to issues such as audience, interior design, use of icons, and illustrations. Our editorial staff is a careful mix of high-tech journalists and experienced book people. Our close contact with the makers of computer products helps ensure accuracy and thorough coverage. Our heavy use of personal computers at every step in production means we can deliver books in the most timely manner.

We are delivering books of high quality at competitive prices on topics customers want. At IDG, we believe in quality, and we have been delivering quality for over 25 years. You'll find no better book on a subject than an IDG book.

John Kilcullen
President and C.E.O.
IDG Books Worldwide, Inc.

IDG Books Worldwide, Inc. is a division of International Data Group. The officers are Patrick J. McGovern, Founder and Board Chairman; Walter Boyd, President; Robert A. Farmer, Vice Chairman. International Data Group's publications include: **ARGENTINA's** Computerworld Argentina, InfoWorld Argentina; **ASIA's** Computerworld Hong Kong, PC World Hong Kong, Computerworld Southeast Asia, PC World Singapore, Computerworld Malaysia, PC World Malaysia; **AUSTRALIA's** Computerworld Australia, Australian PC World, Australian Macworld, Reseller, IDG Sources; **AUSTRIA's** Computerwelt Oesterreich, PC Test; **BRAZIL's** Computerworld, Mundo IBM, Mundo Unix, PC World, Publish; **BULGARIA's** Computerworld Bulgaria, Ediworld, PC World Bulgaria; **CANADA's** Direct Access, Graduate Computerworld, InfoCanada, Network World Canada; **CHILE's** Computerworld, Informatica; **COLUMBIA's** Computerworld Columbia; **CZECH REPUBLIC's** Computerworld Elektronika, PC World; **DENMARK's** CAD/CAM WORLD, Communications World, Computerworld Danmark, Computerworld Focus, Computerworld Uddannelse, Lotus World, Macintosh Produktkatalog, Macworld Danmark, PC World Danmark, PC World Produktguide, Windows World; **EQUADOR's** PC World; **EGYPT's** PC World Middle East; **FINLAND's** Mikro PC, Tietoviikko, Tietoverkko; **FRANCE's** Distributique, GOLDEN MAC, InfoPC, Languages & Systems, Le Guide du Monde Informatique, Le Monde Informatique, Telecoms & Reseaux; **GERMANY's** Computerwoche, Computerwoche Focus, Computerwoche Extra, Computerwoche Karriere, edv aspekte, Information Management, Macwelt, Netzwelt, PC Welt, PC Woche, Publish, Unit; **HUNGARY's** Computerworld SZT, PC World; **INDIA's** Computers & Communications; **ISRAEL's** Computerworld Israel, PC World Israel; **ITALY's** Computerworld Italia, Lotus Magazine, Macworld Italia, Networking Italia, PC World Italia; **JAPAN's** Computerworld Japan, Macworld Japan, SunWorld Japan; **KENYA's** East African Computer News; **KOREA's** Computerworld Korea, Macworld Korea, PC World Korea; **MEXICO's** Compu Edicion, Compu Manufactura, Computacion/Punto de Venta, Computerworld Mexico, MacWorld, Mundo Unix, PC World, Windows; **THE NETHERLANDS'** Computer! Totaal, LAN Magazine, MacWorld Magazine; **NEW ZEALAND's** Computer Listings, Computerworld New Zealand, New Zealand PC World; **NIGERIA's** PC World Africa; **NORWAY's** Computerworld Norge, C/World, Lotusworld Norge, Macworld Norge, Networld, PC World Ekspress, PC World Norge, PC World's Product Guide, Publish World, Student Data, Unix World, Windowsworld, IDG Direct Response; **PANAMA's** PC World; **PERU's** PC World; **PEOPLES REPUBLIC OF CHINA's** China Computerworld, PC World China, Electronics International; **IDG TECH BEIJING's** Electronics New Product World; **IDG SHENZHEN's** Computer News Digest; **PHILLIPPINES'** Computerworld, PC World; **POLAND's** Computerworld Poland, PC World/Komputer; **PORTUGAL's** MacIn; **ROMANIA's** InfoClub Magazine; **RUSSIA's** Computerworld-Moscow, Mir-PC, Sety; **SLOVENIA's** Monitor Magazine; **SOUTH AFRICA's** Computing S.A.; **SPAIN's** Amiga World, Autoedicion, Computerworld Espana, Macworld Espana, NeXTWORLD, PC World Espana, Publish, Sunworld; **SWEDEN's** Attack, ComputerSweden, Corporate Computing, Lokala Natverk/LAN, Lotus World, MAC&PC, Macworld, Mikrodatorn, PC World, Publishing & Design (CAP), DataIngenjoren, Maxi Data, Windows World; **SWITZERLAND's** Computerworld Schweiz, Macworld Schweiz, PC & Workstation; **TAIWAN's** Computerworld Taiwan, Global Computer Express, PC World Taiwan; **THAILAND's** Thai Computerworld; **TURKEY's** Computerworld Monitor, Macworld Turkiye, PC World Turkiye; **UNITED KINGDOM's** Lotus Magazine, Macworld, Sunworld; **UNITED STATES'** AmigaWorld, Cable in the Classroom, CD Review, CIO, Computerworld, Desktop Video World, DOS Resource Guide, Electronic News, Federal Computer Week, Federal Integrator, GamePro, inCider/A+, IDG Books, InfoWorld, InfoWorld Direct, Laser Event, Macworld, Multimedia World, Network World, NeXTWORLD, PC Games, PC World, PC Letter, Publish, SunWorld, SWATPro, Video Event, Video Toaster World; **VENEZUELA's** Computerworld Venezuela, MicroComputerworld Venezuela; **VIETNAM's** PC World Vietnam

# About the Author

Bob Ainsbury is the vice president of software development for TechnoJock Software, Inc., a Houston-based software company specializing in the development of programming libraries for DOS and Windows. He recently was the lead developer on the SmartPak product for 1-2-3 for Windows for Lotus Development Corporation.

Although Bob received a degree in Civil Engineering from the University of Manchester, England, he recently had trouble building a BBQ pit.

When Bob isn't coding or writing, he enjoys shooting pool, playing the drums, and avoiding home-maintenance projects. His favorite film is *The Commitments,* and his favorite clothes are baggy.

# Credits

**Publisher**
David Solomon

**Acquisitions Editor**
Janna Custer

**Managing Editor**
Mary Bednarek

**Project Editors**
Sandra Blackthorn
Laurie Smith
Terrie Lynn Solomon
Diane Graves Steele

**Editors**
Julie King
Darrin Strain
Rebecca Whitney

**Technical Reviewer**
Linda Slovick

**Editorial Assistant**
Patricia R. Reynolds

**Proofreader**
Charles A. Hutchinson

**Production Manager**
Beth J. Baker

**Production Coordinator**
Cindy L. Phipps

**Production Assistant**
Mary A. Briedenbach

**Indexing**
Steve Rath

**Illustrations**
Matt Tyndall Design

**Production**
Beth J. Baker
Cindy L Phipps
Mary A. Briedenbach

**Book Design and Production**
Peppy White
Francette M. Ytsma
Tracy Strub
*(University Graphics, Palo Alto, California)*

# Contents at a Glance

**Foreword** ........ **xxvi**

**Introduction** ........ **1**

**Part I: Secrets for an Effective Beginning** ........ **1**

Chapter 1: Taking an Aerial View of DOS 6 ........ 13
Chapter 2: Mastering the Top DOS Commands ........ 39
Chapter 3: Protecting and Recovering That Irreplaceable Data ........ 71

**Part II: Secret Weapons for Your Hardware** ........ **107**

Chapter 4: Making Your Hard Disk Faster, Bigger, and Safer ........ 109
Chapter 5: Maximizing Memory ........ 159
Chapter 6: Taking Control of the Display and Keyboard ........ 217
Chapter 7: Juggling IRQs, DMAs, Buses, Device Drivers, and Other "What's Its" ........ 241
Chapter 8: Power System Configuration Techniques ........ 261

**Part III: Secrets for Automating DOS** ........ **283**

Chapter 9: Putting DOSKEY to Work ........ 285
Chapter 10: Moving Beyond Basic Batch Files ........ 305
Chapter 11: Demystifying DEBUG ........ 337

**Part IV: The Complete DOS 6 Command Reference** ........ **367**

**Part V: Bob's Better-Than-DOS Shareware Collection** ........ **603**

**Part VI: Appendixes .......................................................................807**
Appendix A: Installing DOS 6 .......................................................................809
Appendix B: DOS Command Quick Reference ..............................................815
Appendix C: ANSI.SYS Quick Reference ........................................................825
Appendix D: Interacting with Workgroups and Networks ...........................831
Appendix E: Additional DOS Information Sources ......................................879

Index .......................................................................................................883
Installing Bob's Better-Than-DOS Shareware Collection Disks ...................952

# Table of Contents

Foreword ........ xxvi

Introduction ........ 1

Part I: Secrets for an Effective Beginning ........ 1

Chapter 1: Taking an Aerial View of DOS 6 ........ 13

Something to Write Home About ........ 13
Configuration Flexibility ........ 14
  Configuration menus ........ 14
  Interruption keys ........ 16
  A new switch for device drivers ........ 17
On-Line Help ........ 17
Advanced Memory Management ........ 19
  Memory optimization tools ........ 20
  EMM386 enhancements ........ 23
Virus Protection ........ 24
Heavy-Duty Disk Utilities ........ 25
  DoubleSpace disk compression ........ 25
  DBLSPACE.BIN ........ 26
  The DBLSPACE command ........ 27
  A real backup program ........ 29
  Disk defragmentation ........ 31
  New undeletion tools ........ 32
  Improved cache performance ........ 33
Windows-Hosted Utilities ........ 33
Laptop Tools ........ 35
Other Goodies ........ 35
What's Gone? ........ 36

**Chapter 2: Mastering the Top DOS Commands .......... 39**
So Many Commands, So Little Time .......... 39
Getting the Best HELP .......... 40
  Running help searches .......... 40
  Solving help conflicts with 4DOS and NDOS .......... 42
  Help tricks for Microsoft C programmers .......... 43
  Using the /? switch .......... 44
  Using FASTHELP .......... 45
  Applications help .......... 46
Kicking the COPY Habit .......... 47
Using the New MOVE Command .......... 49
Deleting Files .......... 50
Zap It with DELTREE .......... 52
DIR — Yes, DIR .......... 53
  Listing subdirectories .......... 53
  Displaying all files, including hidden and system files .......... 53
  Searching for files .......... 54
  Creating a smarter filemask .......... 54
  Searching for a partial directory name .......... 55
  Setting default DIR switches .......... 56
  Shareware alternatives to DIR .......... 57
Directory Navigation Commands .......... 59
A Smarter PATH .......... 61
MSD, the Administrator's Friend .......... 61
Making the DOSSHELL Work .......... 62
  File viewing .......... 64
  Diverse file selection and management .......... 64
  Task switching .......... 67

**Chapter 3: Protecting and Recovering That Irreplaceable Data ... 71**
The DOS 6 Defensive Arsenal .......... 71
Backup Tips, Traps, and Truths .......... 73
  Using the DOS 6 backup software .......... 74
    Comparing MSBACKUP and MWBACKUP .......... 74
    Selecting files to back up .......... 75
      Using point and click .......... 76
      Using include/exclude clauses .......... 76

    - Selecting the backup type ........ 78
    - Special file-selection options ........ 79
    - Make sure that your backup is good ........ 80
    - Optimizing backup speed ........ 81
    - Files created and used by the backup ........ 81
    - Restoring files from a backup ........ 84
    - Backup size limitations ........ 86
  - Backing up DoubleSpace drives ........ 86
  - Developing a backup strategy ........ 86
- "Must Do" Data-Protection Precautions ........ 88
  - Guarding against accidental file deletion ........ 88
    - Installing deletion protection ........ 89
    - Understanding UNDELETE.INI ........ 90
  - DEFRAG — yes, DEFRAG ........ 91
  - Check your disks ........ 92
  - The case of the missing MIRROR ........ 92
- Recovering Lost Files and Data ........ 93
  - Undeleting files ........ 93
  - Recovering from an accidental disk format ........ 95
  - Restoring hard disk partition data ........ 97
- Deliberate Data Destruction ........ 97
- Vital Virus Measures ........ 98
  - The virus phenomenon ........ 98
    - Understanding how viruses spread ........ 99
    - Potential virus damage ........ 100
  - Defending against and identifying viruses ........ 101
    - Scanning for virus infection ........ 101
    - Where did those CHKLIST.CPS files come from? ........ 101
    - Getting rid of those checksum files ........ 102
    - Guarding against destructive viruses ........ 102
  - Removing viruses from an infected system ........ 103
  - Comparing MSAV and MWAV ........ 104
  - On-line virus information ........ 104
- On a Lighter Note ........ 104

Part II: Secret Weapons for Your Hardware .......... 107

Chapter 4: Making Your Hard Disk Faster, Bigger, and Safer ...... 109

The Brawn Behind the Brain .......... 109
The Mechanics of a Hard Disk .......... 110
  Inside a hard disk .......... 110
  The role of disk controllers .......... 112
Data Storage Schemes .......... 113
  Tracks .......... 114
  Sectors .......... 114
  Clusters .......... 115
  How to check your drive characteristics .......... 116
  The file allocation table .......... 117
  Directories .......... 118
  How DOS reads, writes, and deletes files .......... 119
    Reading a file .......... 119
    Deleting a file .......... 119
    Writing a file .......... 120
How to Prepare a New Hard Disk .......... 120
  Low-level formatting .......... 120
  Disk partitioning .......... 121
The Role of CHKDSK .......... 123
Faster, Faster, Faster .......... 126
  Optimizing the interleave .......... 126
    Drive alignment problems .......... 128
    Disk surface deterioration .......... 128
  Improving throughput with caches .......... 129
    Using SMARTDRV .......... 129
      How much memory should you assign to the cache? .......... 129
      Do you need double buffering? .......... 131
      Is delayed write caching appropriate? .......... 132
    Resolving communications problems with SMARTDRV .......... 133
    Using SMARTDRV with DoubleSpace drives .......... 134
    Avoiding bad cluster problems .......... 134
    Using SMARTDRV with CD-ROMs .......... 134

    - Using Fastdisk with Windows ..... 135
    - Using RAM drives ..... 135
    - Using FASTOPEN ..... 136
    - Reviewing the benefits of defragmentation ..... 136
        - Using DEFRAG ..... 137
        - Solving memory problems ..... 139
- Bigger, Bigger, Bigger with DoubleSpace ..... 142
    - Reviewing DoubleSpace mechanics ..... 143
        - Understanding the compressed volume file ..... 144
        - Compressing an entire drive ..... 144
    - Installing a DoubleSpace drive ..... 146
        - Express setup ..... 146
        - Custom setup ..... 146
        - The DoubleSpace files ..... 148
        - The DoubleSpace booting process ..... 148
        - Problems with swap files ..... 149
        - Drive-mounting problems caused by file fragments ..... 150
        - How to defragment DoubleSpace volumes ..... 150
    - Resizing a DoubleSpace drive ..... 151
    - Removing a DoubleSpace drive ..... 152
    - Removing the DoubleSpace device driver from memory ..... 152
    - Changing drive letter assignments ..... 153
    - Understanding drive compression ratios ..... 155
    - Caching compressed drives with SMARTDRV ..... 156
    - Compressing data on floppies ..... 156

**Chapter 5: Maximizing Memory ..... 159**

- Photographs and Memories ..... 159
- The Memory Crunch ..... 160
- Memory Terms, Types, and Tidbits ..... 161
    - ROM vs. RAM ..... 161
    - ROM-BIOS ..... 162
    - Memory hardware ..... 162
        - The DRAM chip ..... 163
        - The SRAM chip ..... 164
        - SIMMs ..... 165

            - The CMOS chip ........ 165
            - Memory speed issues ........ 166
        - Memory addressing ........ 167
            - Hexadecimal numbers ........ 167
            - Segments and offsets ........ 169
        - Major types of memory ........ 171
            - How to check your system ........ 171
            - Base and upper memory ........ 172
            - Expanded memory ........ 175
            - Extended memory ........ 176
            - High memory ........ 182
        - Other memory terms ........ 183
            - Memory managers ........ 183
            - DOS extenders ........ 183
            - Upper memory blocks ........ 184
            - Shadow RAM ........ 184
            - Stealth and BIOS compression ........ 184
            - Virtual memory ........ 184
    - How to Maximize Memory with DOS 6 ........ 185
        - Reviewing the DOS 6 memory arsenal ........ 185
            - Full steam ahead with the 80386 ........ 186
            - Options for 286 systems ........ 186
            - Bad news for 8086 owners ........ 186
            - Alternatives for 8086 and 286 owners ........ 186
        - Reviewing your memory settings with MEM ........ 187
        - Using the big gainers ........ 189
            - Load DOS high ........ 189
            - Use the UMBs ........ 190
        - Shifting from RAM cram to UMB cram ........ 192
            - Identifying potential UMB space ........ 192
            - Controlling load order and location ........ 193
            - Tweaking UMB memory allocation ........ 194
        - Loading DBLSPACE.BIN high ........ 195
        - Letting MEMMAKER do the work ........ 195
            - Looking at MEMMAKER in action ........ 196
            - Running MEMMAKER ........ 198
                - Optimize upper memory for use with Windows ........ 200
                - Use the monochrome region ........ 201

    Understanding how MEMMAKER works .......... 201
    Helping MEMMAKER succeed .......... 204
      Avoid using a multiple configuration menu in CONFIG.SYS .......... 204
      Don't call programs from other batch files .......... 205
      Don't call programs in IF statements .......... 205
      Exclude some programs from analysis and execution .......... 206
  Saving every drop of base memory .......... 207
    Minimizing the environment size .......... 208
    Tuning the FILES statement .......... 209
    Tuning the BUFFERS statement .......... 209
    Optimizing the STACKS command .......... 209
    Minimizing the FCBS allocation .......... 210
    Being frugal with LASTDRIVE .......... 210
    Minimize MSCDEX base memory consumption .......... 210
  Maximizing extended memory .......... 210
  Managing expanded memory .......... 211
  Solving memory conflicts .......... 212
    Running MEMMAKER locks up the system .......... 212
    The new configuration locks up the system .......... 213
    A program causes system lockup .......... 214
    A program runs more slowly .......... 214
    The system doesn't respond to Ctrl-Alt-Del .......... 214

**Chapter 6: Taking Control of the Display and Keyboard .......... 217**
We Want a Color TV! We Want a Color TV! .......... 217
Reviewing ANSI.SYS Techniques .......... 218
  Sending instructions to ANSI.SYS .......... 218
    Using batch files .......... 219
    Using the TYPE command .......... 220
    Using PROMPT .......... 220
    Sending multiple escape sequences .......... 221
  Determining whether ANSI.SYS is loaded .......... 222
Jazzing Up the Display .......... 223
  Setting foreground and background colors .......... 224
  Setting display attributes .......... 225
  Positioning the cursor .......... 226
  Pulling it all together .......... 227
  Creating fancy prompts .......... 230

Customizing the Keyboard ........ 232
  Key codes ........ 232
  How to remap keys ........ 237
  How to assign strings or commands to keys ........ 237
Going Beyond ANSI.SYS ........ 239

**Chapter 7: Juggling IRQs, DMAs, Buses, Device Drivers, and Other "What's Its" ........ 241**

Looking under the Hood ........ 241
Understanding Interrupts and IRQ Conflicts ........ 242
  Interrupting the microprocessor ........ 242
  Interrupt vectors ........ 243
  Interrupt request lines ........ 244
  Reviewing your IRQ assignments ........ 246
Understanding DMA ........ 247
Which Bus Is Best: ISA, MCA, or EISA? ........ 248
  The role of the bus ........ 249
  The ISA bus ........ 249
  The MCA bus ........ 251
  The EISA bus ........ 252
  Recent bus innovations ........ 255
    The local bus ........ 255
    SCSI, the UnBus ........ 256
    What's next? ........ 257
Deciphering Device Drivers ........ 257

**Chapter 8: Power System Configuration Techniques ........ 261**

Interrupting and Controlling the Boot Cycle ........ 261
  Using the clean-start key ........ 262
  Using the interactive start key ........ 263
  Forcing a [Y/N] prompt ........ 264
  Controlling start keys with the SWITCHES command ........ 264
Creating Multiboot Configurations ........ 265
  Defining menus ........ 266
    Setting menu defaults ........ 268

- Changing menu colors ... 268
- Changing menu headings ... 269
- Combining common commands ... 269
- Using multimenu configurations ... 272
- Providing an escape option ... 274
- Prompting for user confirmation ... 275
- Configuring AUTOEXEC.BAT ... 276
- Checking CONFIG.SYS ... 278
- Checking AUTOEXEC.BAT ... 281

**Part III: Secrets for Automating DOS ... 283**

**Chapter 9: Putting DOSKEY to Work ... 285**

- Installing DOSKEY ... 285
  - Setting Insert Mode ... 286
  - Setting the buffer size ... 286
- Reinstalling DOSKEY ... 287
- Using DOSKEY as a Command-Line Manager ... 287
  - Editing the command line ... 287
  - Using the command history ... 288
  - My top five (forgotten) DOSKEY commands ... 288
    - Selecting commands by number ... 288
    - Using characters to select commands ... 289
    - Saving a command without executing it ... 289
    - Executing two or more commands at a time ... 290
    - Erasing the entire command history ... 290
- Using DOSKEY Macros ... 290
  - Accessing command-line switches ... 291
  - Piping and redirection with DOSKEY ... 292
  - Entering multiple commands ... 293
  - Building a macro builder ... 293
- Comparing DOSKEY Macros to Batch Files ... 294
- My Favorite DOSKEY Macro Tricks ... 295
  - Question-mark help ... 295
  - Replacing, enhancing, and disabling DOS commands ... 296

Intercepting drive changes ........ 297
Supporting multiple filenames ........ 297
Extending the search path ........ 298
And some quickies to finish ........ 298
Going Beyond DOSKEY with ANARKEY ........ 299
Command-line editing ........ 299
Environment editing ........ 300
History lists ........ 301
Macros ........ 302
The MegaKey and MenuKey shortcuts ........ 302

**Chapter 10: Moving Beyond Basic Batch Files ........ 305**

The Tools of the Trade ........ 305
A batch file command summary ........ 306
It's about time we had a choice ........ 308
Editor tips ........ 311
Power Techniques That Use Redirection and Pipes ........ 313
Just say yes ........ 314
Just be quiet ........ 315
Displaying command output selectively ........ 316
Environment Innovations ........ 317
Temporarily changing the search path ........ 318
Managing the TEMP variable ........ 320
Saving and restoring the environment ........ 320
Uppercasing input ........ 321
Dates, Times, and Batch Files ........ 323
Waiting for a while ........ 323
What day of the week is it? ........ 324
What time is it? ........ 324
Waiting for a specific time ........ 325
Running batch files once a day ........ 327
A Batch File Potpourri ........ 328
Does a directory exist? ........ 328
Checking for wildcards ........ 329
Checking the DOS version ........ 330
Checking for TSRs ........ 330
Saving the current directory ........ 331

Using Batch Files with Windows 3.1 ....... 332
  Controlling DOS session settings ....... 332
  Exploiting WINSTART.BAT ....... 333
  Using the new command /K switch ....... 334
  Checking for Windows 3.1 ....... 334
Batch File Enhancers ....... 335
  Adding pizzazz with Batutil ....... 335
  Putting batch files in overdrive with TURBOBAT ....... 336

## Chapter 11: Demystifying DEBUG ....... 337

Understanding VCRs, Understanding DEBUG ....... 337
Getting Acquainted with DEBUG Editing Capabilities ....... 338
Using DEBUG Commands ....... 339
  DEBUG command reference ....... 339
    The ? (help) command ....... 340
    The A command ....... 340
    The C command ....... 341
    The D command ....... 341
    The E command ....... 342
    The F command ....... 342
    The G command ....... 342
    The H command ....... 342
    The I command ....... 343
    The L command ....... 343
    The M command ....... 343
    The N command ....... 344
    The O command ....... 344
    The P command ....... 344
    The Q command ....... 344
    The R command ....... 344
    The S command ....... 345
    The T command ....... 345
    The U command ....... 345
    The W command ....... 345
    The XA command ....... 346
    The XD command ....... 346
    The XM command ....... 346
    The XS command ....... 346

How to enter DEBUG commands ........ 346
Taking a test drive ........ 347
Creating ASCII script files ........ 349
Patching Programs with DEBUG ........ 350
Changing the CHOICE command ........ 350
Patching the default EDIT filemask ........ 352
Creating Useful Utilities with DEBUG ........ 355
Returning to DOS and setting an ERRORLEVEL ........ 355
Working with dates ........ 356
Writing keyboard management utilities ........ 358
Checking and changing the state of special keys ........ 358
Rebooting the PC ........ 361
Waiting for a function key ........ 361
Creating disk and drive utilities ........ 363
Abort, Retry, Fail? Not! ........ 363
A script to determine the boot drive ........ 365

**Part IV: The Complete DOS 6 Command Reference ........ 367**

**Part V: Bob's Better-Than-DOS Shareware Collection ........ 603**

**Part VI: Appendixes ........ 807**

**Appendix A: Installing DOS 6 ........ 809**
Upgrading to DOS 6 ........ 809
Installing to Floppies ........ 810
Installing a Minimum Configuration ........ 811
Installing the Windows Components of DOS 6 ........ 812
Installing DOS 6 Manually ........ 812
Installing to a Nonboot Drive ........ 812
Uninstalling DOS 6 ........ 813

**Appendix B: DOS Command Quick Reference ........ 815**
Standard DOS 6 Files ........ 815
Files Installed Only When an Older Version of the File Is Present on the System ........ 822
Files Used by SETUP but Not Installed ........ 823
Old Files Available on the DOS 6 Supplemental Disk ........ 823

**Appendix C: ANSI.SYS Quick Reference ........ 825**
Color Display Codes Used with Esc[codem ........ 825
Display Attribute Codes Used with Esc[codem ........ 826
Text Erasing Codes ........ 826
Cursor-Control Codes ........ 826
Display Mode Control Codes ........ 827
Keyboard Control Codes ........ 828

**Appendix D: Interacting with Workgroups and Networks ........ 831**
The InterLink Connection ........ 831
  Wiring the two computers together ........ 831
  Establishing an InterLink connection ........ 832
    How to install INTERLNK ........ 832
    How to install INTERSVR ........ 833
    Three easy steps to successful linking ........ 833
  Using server resources ........ 834
  Understanding InterLink limitations ........ 835
Workgroup Wizardry ........ 835
  Installing the network software ........ 836
  Reviewing network commands ........ 838
  Connecting to a network ........ 838
    Network entries in CONFIG.SYS ........ 839
    How to log on to the network ........ 841
    Password management ........ 842
  Using network resources ........ 843
    Using the Pop-Up utility ........ 843
      Loading and running Pop-Up ........ 843
      Connecting to directories ........ 845
      Connecting to shared printers ........ 846
    Using the command-line utilities ........ 846
      Viewing resources with NET VIEW ........ 846
      Accessing remote directories and printers ........ 847
      Reviewing directory examples ........ 848
      Reviewing printer examples ........ 849
    Accessing CD-ROMs ........ 850
Mail Secrets ........ 850
  Installing Mail ........ 850
  Running Microsoft Mail ........ 852
  Locating your mail files ........ 852

Receiving new mail .......... 853
Forgetting your password or mailbox name .......... 855
Workgroup Connection Command Reference .......... 855
MAIL .......... 856
Description .......... 856
Usage .......... 856
Options and switches .......... 856
Remarks .......... 858
Examples .......... 858
Additional information .......... 858
MICRO .......... 858
Description .......... 858
Usage .......... 858
Options and switches .......... 858
Remarks .......... 859
Examples .......... 860
Additional information .......... 860
NET .......... 860
Description .......... 860
NET CONFIG .......... 861
Description .......... 861
Usage .......... 861
Options and switches .......... 861
Remarks .......... 862
Example .......... 862
NET LOGOFF .......... 862
Description .......... 862
Usage .......... 862
Options and switches .......... 862
Remarks .......... 862
Example .......... 863
NET LOGON .......... 863
Description .......... 863
Usage .......... 863
Options and switches .......... 863
Remarks .......... 863
Examples .......... 864
Additional information .......... 864

NET PASSWORD ....865
  Description ....865
  Usage ....865
  Options and switches ....865
  Remarks ....865
  Examples ....865
  Extended use ....866
NET PRINT ....866
  Description ....866
  Usage ....866
  Options and switches ....866
  Remarks ....867
  Examples ....867
  Additional information ....868
NET START ....868
  Description ....868
  Usage ....868
  Options and switches ....868
  Remarks ....869
  Examples ....869
  Additional information ....870
NET STOP ....870
  Description ....870
  Usage ....870
  Options and switches ....870
  Remarks ....871
  Examples ....871
NET TIME ....871
  Description ....871
  Usage ....871
  Options and switches ....871
  Remarks ....872
  Example ....872
NET USE ....872
  Description ....872
  Usage ....872
  Options and switches ....872
  Remarks ....874
  Examples ....874
  Additional information ....875

NET VER ....................876
  - Description ....................876
  - Usage ....................876
  - Options and switches ....................876
  - Remarks ....................876

NET VIEW ....................876
  - Description ....................876
  - Usage ....................876
  - Options and switches ....................876
  - Remarks ....................877
  - Examples ....................877

**Appendix E: Additional DOS Information Sources ....................879**
- Microsoft Telephone Support ....................879
- Magazines and Periodicals ....................879
- PC User Groups ....................880
- On-Line Sources ....................880
- Information for Users with Disabilities ....................881

**Index ....................883**

**Disclaimer page ....................952**

# Introduction

DOS 6 began shipping nearly twelve years after the introduction of the first version of DOS in August 1981. During this time, PCs have ascended from humble beginnings to today's systems sporting 128 megabytes of memory and gigabytes of storage space.

The only significant changes in DOS from August 1981 to November 1988 (DOS 4.0) were hardware related, such as adding support of 3½-inch disk drives and networks. DOS 5 was the first version of the operating system that provided significant end-user enhancements — a full-screen editor ousted EDLIN, for example, and DOS 5 came with a reasonable help system to provide users with on-line help.

Interestingly, Microsoft's improved *user orientation* on DOS upgrades came at a time when another vendor, Digital Research (now part of Novell Inc.), was having significant market success with a competitive *friendlier* operating system, DR DOS. As is often the case in the computer business, we, the users, have benefited from vendor competition.

Microsoft has continued its user focus with the introduction of DOS 6. There are plenty of all-new facilities, such as virus protection, sophisticated memory optimization, and on-the-fly disk compression, as well as major improvements to some old war horses like BACKUP.

DOS 6 is packed with new features, utilities, and enhancements, and every DOS user should seriously consider upgrading. For the first time ever, I really enjoyed upgrading my version of DOS.

## Why Another DOS Book?

The shelves in computer bookstores are weighed down with DOS books, but *DOS 6 SECRETS* is different from the rest of them.

*DOS 6 SECRETS* goes beyond a command reference (although a very rich and exhaustive command reference is included); in this book, you are given guidance as to the *importance* and *value* of each command.

*DOS 6 SECRETS* doesn't go into excruciating detail on obscure and little-used features. You won't find twenty pages discussing the various EDLIN commands, and CTTY is given a lot less coverage than CHOICE. Nor will you have to trudge through lengthy explanations of well-known concepts, such as how to create and change directories or what a filename extension is.

This book shows you how to use DOS in the real world, using practical examples and insights. Throughout the editorial process, everybody involved in *DOS 6 SECRETS* has strived to make sure that the information is not just accurate but that it is *relevant* to the PC user.

## The Audience for This Book

*DOS 6 SECRETS* is written for a broad base of PC users, ranging from the infrequent user all the way to a DOS Jedi Master. I do, however, assume that you are familiar with the basic tenets of using DOS, such as editing files, creating and managing directory hierarchies, and so on. If you have never used a PC before or you are just in the early learning stages, I recommend that you first cut your teeth on the excellent IDG book *DOS For Dummies,* 2nd Edition. The target audience for *DOS 6 SECRETS* is quite simply current DOS users who want to learn more.

If you have been using a computer for some time but feel that you are not getting the best results, or you have used a computer that has been installed and set up by someone else and you want to learn more, then *DOS 6 SECRETS* is for you.

This book is also valuable to long-time DOS users who first started using DOS in the early 1980s. Very often these users are experts in the commands that have been around since DOS 1.0. Typically, they have used commands like COPY, DEL, and DIR hundreds of thousands of times. The trouble is, many of the original commands have been improved since the mid 1980s. For example, the old timers know that the /W and /P switches can be used to improve the readability of the output from DIR, but how many realize that the /A switch can be used to display all files — including system and hidden files — or that the /S switch can be used to search all subdirectories?

Finally, if you are an experienced DOS 5 user, remember that DOS 6 is packed with new, very useful features, such as virus protection, on-the-fly disk compression, multiple boot-up configurations, and much more. Particular care has been taken to ensure that these new features are covered in depth.

Armed with *DOS 6 SECRETS,* you'll soon become a DOS 6 Jedi Master.

## How This Book Is Organized

The book is organized into five main parts. The first two parts concentrate on how to use the primary DOS commands and how to get the most from your hardware. The third part explains how programming enables complex operations to be reduced to single commands and how routine tasks can be automated. Part IV is an exhaustive command reference to each and every command and option available in DOS. DOS, however, doesn't have all the answers, so the fifth and final part examines a wealth of Shareware and public domain programs that further empower the user. All the programs and utilities discussed in Part V are included on the Bob's Better-Than-DOS Shareware Collection disks.

The appendixes include information on how to install DOS 6, along with at-a-glance command summaries, and information about other ways to increase your DOS knowledge.

## Part I: Secrets for an Effective Beginning

The first part of the book focuses on those features of DOS that are mostly used on a regular and frequent basis. The *eighty/twenty rule* states that 80 percent of the time you will use 20 percent of the DOS commands. The primary goal of Part I is to make sure that you are an expert in the most-used 20 percent. You will also learn how to use DOS efficiently and make DOS work for you — and you will pick up good habits to avoid data loss.

**Chapter 1, "Taking an Aerial View of DOS 6,"** provides a flight over the DOS 6 terrain. This chapter discusses each major component of the operating system with particular emphasis on the new features introduced in DOS 6, such as virus protection and data compression. After reading this chapter, you will be ready to enter the DOS 6 inner circle.

**Chapter 2, "Mastering the Top DOS Commands,"** offers an in-depth discussion of the everyday DOS commands. This chapter describes how to get the most from the most popular and frequently used commands like XCOPY and DOSKEY. Some of the best Shareware alternatives are also introduced.

**Chapter 3, "Protecting and Recovering That Irreplaceable Data,"** is a must for every PC user. The data stored in your computer is valuable — probably more valuable than you realize. As my sister used to say, "You'll miss me when I'm gone!" This chapter explains all the measures you should take to ensure that your data is safe and secure. You will learn how specific preventive measures can alert you to problems before they occur as well as how to recover files after disaster has struck. No chapter on data security would be complete without a discussion of backups. This chapter provides detailed guidance on how to establish an efficient, nondisruptive regimen to keep valuable file backups.

## Part II: Secret Weapons for Your Hardware

Most of us want more for less. In the case of your PC, you may already have more but don't know it. The second part of *DOS 6 SECRETS* focuses on how to fine-tune DOS so that you really can leverage the performance of your computer hardware. A DOS Jedi must use the power of the hardware. Part II explains how to find and use every nook and cranny of your memory and hard disk, along with ways to jazz up the boring gray and black DOS screen.

**Chapter 4, "Making Your Hard Disk Faster, Bigger, and Safer,"** as the title suggests, explains how to squeeze every last drop of performance and capacity from your hard disk. Improving hard disk performance is the single most effective way of improving your system's overall performance, and this chapter shows you how. You'll learn how a hard disk physically works as well as how DOS and the disk controller maintain detailed records about every file's data. This chapter also explains how to double your storage capacity with the all-new DoubleSpace utility and how to dramatically improve disk performance using a disk cache.

**Chapter 5, "Maximizing Memory,"** starts by explaining all the various memory issues in plain terms. You will learn how the memory *warehouse* stores base, high, upper, extended, and expanded memory. The chapter goes on to explain how you can tune your system using the all new MEMMAKER program to dramatically increase the memory available to your applications.

**Chapter 6, "Taking Control of the Display and Keyboard,"** shows you how ANSI.SYS can be used to give the DOS display a much-needed facelift. This chapter also discusses techniques for customizing the keyboard.

**Chapter 7, "Juggling IRQs, DMAs, Buses, Device Drivers, and Other 'What's Its',"** explains, in layman's terms, important hardware facts that can help you to overcome conflicts between various devices. For example, your fax board might fight with your scanner for the CPU's attention. This chapter shows you why these conflicts occur and what steps can be taken to resolve them.

**Chapter 8, "Power System Configuration Techniques,"** concludes Part II and explains how to organize your CONFIG.SYS and AUTOEXEC.BAT file to meet all your requirements. Techniques for building boot-up menus (allowing you to choose from more than one configuration set) are presented, together with a checklist of all the items you should include in your boot-up files.

## Part III: Secrets for Automating DOS

Computers are meant to automate and simplify tasks. If you find yourself frequently repeating a series of commands or writing cheat-sheet instructions on how to do specific operations, you ought to be thinking of a better way. There is one. Part III explains how to use DOS's own tools to automate and program your computer.

**Chapter 9, "Putting DOSKEY to Work,"** explains how DOSKEY enhances command-line input and shows how DOSKEY's macro facility provides a powerful and easy programming tool. You'll learn how to integrate batch files with DOSKEY, how to save your favorite DOSKEY settings, and how to avoid some of the DOSKEY pitfalls.

**Chapter 10, "Moving Beyond Basic Batch Files,"** shows how to leverage the most power from batch files. You will learn how to prompt the user for input, how to manage the DOS environment, and how to use the batch file programming statements like IF and FOR. The chapter concludes with a look at a variety of batch files that should be in every DOS user's arsenal.

**Chapter 11, "Demystifying DEBUG,"** shows how to make the terse and grumpy DEBUG program work for you. You'll learn how to build very small, compact programs to perform a variety of low-level functions, such as rebooting the system, checking whether a disk is in the drive, and more.

## Part IV: The Complete DOS 6 Command Reference

This part of the book provides a complete and thorough reference to all the DOS device drivers and programs. Part IV goes beyond a clinical command reference and provides explanations of how and when each command should be used. Each command is awarded a score on "Bob's Scorecard" indicating its value and merit. The awards range from 10, indicating a very valuable command, to 0, which indicates the command should be locked away in the basement.

## Part V: Bob's Better-Than-DOS Shareware Collection

As good as DOS 6 is, there are still many areas where third-party utilities are superior. For example, consider the EDIT command. There is no doubt that EDIT is a great improvement on EDLIN, but it is still lacking in some areas. The Shareware program QEDIT is far superior to EDIT, providing multifile editing, column manipulation for easy indenting, file merging, a macro language, and much more.

*DOS 6 SECRETS* includes two disks packed with the cream of the Shareware crop. Part V explains how to install the Shareware programs and includes thorough documentation on each program.

## Part VI: Appendixes

**Appendix A, "Installing DOS 6,"** provides detailed instructions on how to install and uninstall DOS 6.

**Appendix B, "DOS Command Quick Reference,"** includes a handy quick reference of all the DOS commands and device drivers.

**Appendix C, "ANSI.SYS Quick Reference,"** provides quick references for ANSI.SYS.

**Appendix D, "Interacting with Workgroups and Networks,"** explains how the InterLink commands can be used to connect two PCs and how the Workgroup Connection software can be used with DOS 6 to access devices such as disks, CD-ROMs, and printers on other computers.

Finally, **Appendix E, "Additional DOS Information Sources,"** suggests other sources where you can gain further information on DOS and Shareware in general.

## Versions of DOS

From a marketing perspective, it might have been desirable to write this book in such a style that it could be read by users of any version of DOS. But who cares about EDLIN now that there is EDIT or about BACKUP now that there is MSBACKUP! Consequently, this book concentrates on the features available in DOS 5 and, especially, DOS 6. The features newly implemented in DOS 6 are clearly marked with an icon so that users of earlier versions of DOS will know what they are missing.

If you don't know which version of DOS you are using, enter the command `ver`; DOS will respond with the version number. To be honest, if you are running a system with an earlier version of DOS, you really ought to upgrade to DOS 6.

## What's on the Disks?

Good though DOS 6 is, there are still some excellent Shareware programs that run rings around their DOS counterparts. *Bob's Better-Than-DOS Shareware Collection* disks are packed full of the best Shareware utilities available today. In addition to programs which are better than COPY, MOVE, FORMAT, and DIR, you'll find programs for duplicating disks, managing your hard disk, listing files, compressing files and directories, compiling batch files, and much, much more.

Having completed the book, we didn't simply throw a few programs onto the disks and bundle them with the book. The Shareware programs have been carefully selected because, in my opinion, they provide better functionality and performance than the DOS 6 commands. Throughout the text, you will find references to these Shareware programs with clear explanations as to why they have an edge over DOS.

Remember that most of these utilities are Shareware and are not free. Copies have been included on the disks so that you can evaluate them. Use them one or two times; if you want to continue to use them, you should register them. Quite simply, Shareware is a try-before-you-buy honor system. For further information on the Shareware system, along with comprehensive documentation of each program, refer to Part V.

Complete installation instructions appear on the last page of the book.

## Conventions Used in This Book

Important facts, tips, and suggestions appear in the text with eye-catching icons in the margin. Here is a comprehensive list of the icons scattered throughout the book:

| | |
|---|---|
| NOTE | This icon highlights a special point of interest about the topic under discussion. |
| TIP | This icon points to a useful hint that might save you time or trouble. |
| CAUTION | This icon alerts you to potential problems. |
| CROSS REFERENCE | This icon, which is used in Part IV, "The Complete DOS 6 Command Reference," points to a more complete discussion of the commands in the chapters of the book and also points you to related commands. |

GEM This icon highlights information that gives you some special insight into the topic under discussion.

DISK This icon alerts you that the item under discussion is included on Bob's Better-Than-DOS Shareware Collection disks packaged with the book.

DOS 6 This icon alerts you to a subject that is new to DOS 6.

UNDOCUMENTED FEATURE This icon points to information that is not available in the DOS documentation.

ERROR MESSAGES DECODED This icon highlights explanations of error messages that DOS displays when problems occur.

TECHNICAL STUFF This icon points to technical discussions that you may want to read to be a well-rounded DOS Jedi.

👎 This icon, which is used in Chapter 1, identifies text that discusses some of DOS 6's trouble spots.

## Sidebars

*Sidebars* are often used to discuss an item in greater detail. For example, one sidebar explains how file undeletion actually works, whereas the main text just explains how to undelete a file. Other times, the sidebars are more anecdotal and provide some general background material, such as the history of the UART chip. If you don't want to delve too deeply into a subject, just read the body of the text and skip most of the sidebars.

# How Commands Are Explained

Listed next are some of the conventions used to identify commands and keystrokes:

- All DOS commands mentioned within a paragraph are displayed in UPPERCASE letters.
- Key combinations in which you hold down one key and press another are shown in hyphenated format. Ctrl-F10, for example, indicates that you hold down the Ctrl key while you press F10.
- Key combinations pressed in sequence are separated by commas (Ctrl-PgUp, down arrow, down arrow).
- *Italic* type is used for new terms or emphasis, as well as variable items in command syntax lines.
- For commands selected from DOS-based menus, **bold** is used to indicate the letter the user presses to choose the command (**O**ptions⇨Help **P**ath). For commands selected from Windows-based menus, an underline is used to indicate the letter the user presses to choose the command (File⇨New⇨Group).
- On-screen information and batch file listings appear in a `special typeface.`
- Literal commands that you type appear in a **`bold special typeface.`**

Most of the DOS commands can accept varying numbers and types of options or switches. For example, when you execute the change directory command, CD, with no options, DOS displays the default directory. However, when you specify a directory path, DOS tries to change the default directory to the specified path.

Although this makes for a flexible command mechanism, it is a pain to document. A special syntax is used to define each command and its switches. For example, the syntax for the CD command is defined as follows:

```
cd [drive:][path] or [..]
```

I admit that the syntax is daunting at first, but once understood, it provides a very convenient and concise way to document a command.

At the beginning of Part IV you will find a thorough explanation of the syntax. In brief, optional commands are included in square brackets, and required command components are not in brackets. If all the options and switches are themselves surrounded by square brackets, the command can be executed without any of them, as is the case in the preceding CD example. Note also that variable elements appear in italics.

***Note:*** This syntax is very similar to that used in the DOS 6 on-line help.

## Give Us Feedback

The best products in the computer marketplace are the ones that have been molded and shaped by user feedback. And so it is for computer books. We sincerely solicit your feedback and criticism. Please write to me in care of IDG Books at the following address:

IDG Books Worldwide
3250 N. Post Road, Ste. 140
Indianapolis, IN 46226

Or you can contact me through CompuServe. My CompuServe ID is 74017,227.

By the way, I'd especially like to hear *your* DOS secrets.

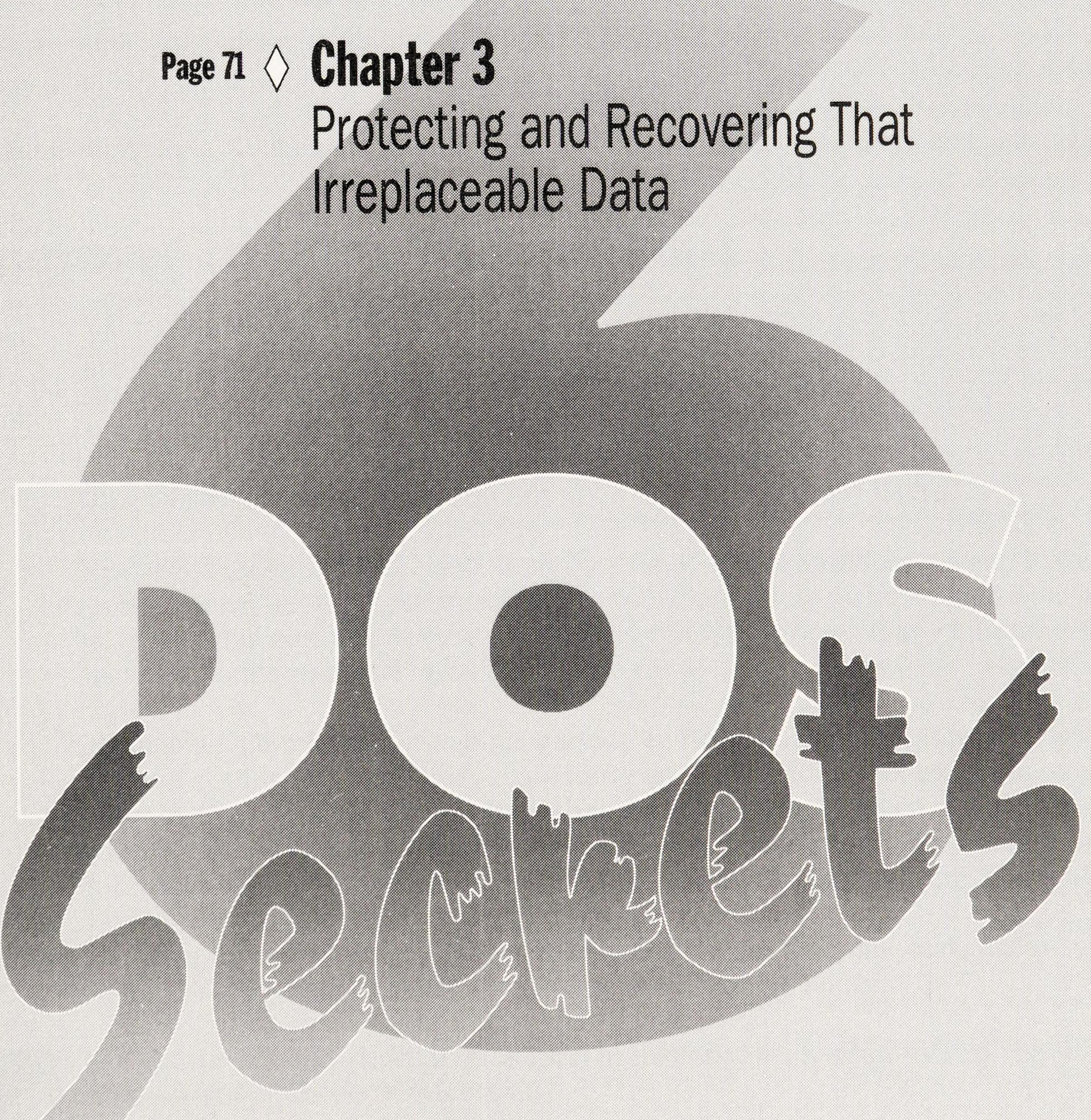
Enough preliminaries, let's get busy.

# Part I:
# Secrets for an Effective Beginning

Page 13 ◊ **Chapter 1**
Taking an Aerial View of DOS 6

Page 39 ◊ **Chapter 2**
Mastering the Top DOS Commands

Page 71 ◊ **Chapter 3**
Protecting and Recovering That Irreplaceable Data



## Chapter 1

# Taking an Aerial View of DOS 6

**In This Chapter**

- Introduction to the new and improved DOS
- A look at some DOS 6 drawbacks
- A quick guide to using important DOS 6 commands and programs

## Something to Write Home About

Past upgrades of DOS left me cold. They seemed to offer few real improvements — just a new switch for FORMAT here, support for 3 ½-inch drives there, and maybe an occasional extra switch. Every now and then, an upgrade provided something interesting, such as a full-screen editor or the ability to load DOS high, but more often than not, upgrades were incremental and often hardware related. They were necessary, no doubt, but not a lot to excite most users.

The DOS 6 upgrade is different. It offers a bevy of new applications and features, such as a real backup system, anti-virus tools, on-the-fly disk compression, and a lot more. This new, user-oriented version of DOS embraces many functions previously left to the third-party utilities market.

Of course, DOS 6 does have some warts. One flaw is that the keystrokes used to perform certain functions vary depending on the situation. For example, you sometimes press Alt-X to quit, but other times you press Ctrl-F4. Another blemish is that programs such as XCOPY still don't take advantage of extended memory. But despite these and other drawbacks, DOS 6 is well worth writing home about.

This chapter provides a summary of the most interesting new features introduced in DOS 6 and also discusses some of its trouble spots.

The special thumbs-down icon is used in this chapter to identify text that discusses some of those DOS 6 warts!

## Configuration Flexibility

Several new enhancements make CONFIG.SYS far more flexible than ever before. The most-publicized improvement is the support of new start-up menu commands. Other enhancements include new keys to interrupt the normal boot cycle and control command execution, plus a new ? switch to force a confirmation prompt before a device driver is loaded.

### Configuration menus

With so many add-on hardware devices installed in PCs these days, many users resort to multiple configuration files. One CONFIG.SYS and AUTOEXEC.BAT pair might be used to install a scanner and digitizing tablet for desktop publishing, and another pair might load a mouse and log on to a network for normal day-to-day activities. Prior to DOS 6, users either had to purchase a third-party utility that supported multiple configurations or write batch files to copy the appropriate configuration files to the root directory and reboot.

Now, DOS provides special menu commands that enable you to place all your diverse configuration commands in a single copy of CONFIG.SYS. When the system boots, you are presented with a list of menu choices. You select a menu option, and DOS executes the specific commands associated with that option. The following is a typical boot menu:

```
MS-DOS 6 Startup Menu
======================

    1. Clean (Most Memory)
    2. Laptop Connection
    3. Standard
```

You can control the format of the menu using the new commands described in Table 1-1.

The following CONFIG.SYS file created the menu shown earlier:

```
[menu]
menuitem=Clean,Clean (Most Memory)
menuitem=Interlink,Laptop Connection
menuitem=Normal,Standard

[common]
BUFFERS=15,0
FILES=30
LASTDRIVE=G
```

**Table 1-1 New Menu Commands**

| *Command* | *Description* |
|---|---|
| [MENU] | Marks the beginning of the menu definition block. |
| MENUITEM | Creates an item on the menu and identifies the name of the associated configuration block. When the user selects the menu item, DOS executes the commands in the specified configuration block. |
| SUBMENU | Creates a branch to a secondary menu, thereby providing nested or multilevel configuration menus. |
| [*BLOCKNAME*] | Every blockname specified with the MENUITEM and SUBMENU commands must have a corresponding block defined. The commands following the menu block are executed when the user selects the corresponding menu item. |
| MENUDEFAULT | Sets the default item in a menu and, optionally, the number of seconds DOS waits for user input before applying the default. |
| INCLUDE | Includes the commands from one configuration block in another. |
| MENUCOLOR | Sets the display colors for the menu. |

```
[Clean]
FCBS=4,0

[Normal]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB
FCBS=4,0
SHELL=C:\DOS\COMMAND.COM C:\DOS\  /p
DEVICEHIGH /L:1,47296 =C:\DOS\DBLSPACE.SYS

[InterLink]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB
DEVICE=C:\DOS\INTERLNK.EXE /DRIVES:5
```

All configuration commands placed in a special block called [COMMON] are executed no matter which option is selected.

DOS 6 provides no specific enhancements to AUTOEXEC.BAT that are related to the multiple configuration feature, but it does give you an easy way to synchronize AUTOEXEC.BAT and CONFIG.SYS. When you make a selection from the start-up menu, DOS creates an environment variable with the name CONFIG and assigns it the name of the selected menu option. By testing the value of the CONFIG variable, you can branch to a specific part of AUTOEXEC.BAT as follows:

```
goto %CONFIG%
```

Refer to Chapter 8 for a thorough discussion of the new multiple configuration feature.

## Interruption keys

If you ever pounded on the Ctrl-Break key to try to stop DOS from executing the start-up commands during system boot, you will be pleased to hear that DOS 6 provides you with two special keys that make it easy to suspend, stop, or selectively execute the start-up commands.

When the system boots, DOS displays the following message for approximately 2 seconds:

```
Starting MS-DOS...
```

If you press F5 (or hold down the Shift key) while this message is displayed, DOS ignores all the commands in CONFIG.SYS and AUTOEXEC.BAT and immediately displays the DOS prompt. This procedure is referred to as *clean booting*.

If you want to execute some, but not all, start-up commands, you can press F8 when the `Starting MS-DOS...` message is displayed. DOS then displays each command in turn and prompts you to press Y or N to indicate whether the command should be executed. After all the CONFIG.SYS commands are processed, DOS asks for confirmation before executing AUTOEXEC.BAT.

If you step through some commands and then want to exit without processing any more CONFIG.SYS commands, you can take a shortcut: Press the F5 key while DOS is prompting for confirmation. Similarly, if you are stepping through the CONFIG.SYS commands and decide that you want to execute all remaining commands along with AUTOEXEC.BAT, you can simply press Esc. DOS stops prompting you for confirmation and executes all remaining commands.

GEM

The SWITCHES configuration command (which is used in CONFIG.SYS) was enhanced so that you can disable the interruption keys, ensuring that all CONFIG.SYS commands are executed without user intervention. The /F switch instructs DOS to proceed without pausing for 2 seconds when the message `Starting MS-DOS...` is displayed. The /N switch instructs DOS to ignore the interruption keys F5, Shift, and F8.

## A new switch for device drivers

You can instruct DOS 6 to prompt you for confirmation before it executes statements in CONFIG.SYS. You can force a `[Y/N]` confirmation prompt on any statement (except switches) by inserting a question-mark character (?) before the equals character (=) in the statement. After you make this change, DOS always prompts you for confirmation before executing the statement.

Suppose, for example, that sometimes you load a device driver for a CD-ROM but other times you don't. The following statement installs the device driver with the new ? switch:

```
device?=c:\proaudio\tslcdr.sys /d:MVCD001 /r /p:3
```

If you use such a statement, DOS prompts you for confirmation before loading the device driver. I wish the same feature were available in batch files.

Further details about the new boot interruption keys and the enhancements to SWITCHES are provided in Chapter 8.

## On-Line Help

DOS 5 introduced the notion of an on-line help system for DOS commands. When you entered the HELP command and a command name, DOS displayed a single screen of information about the specified command.

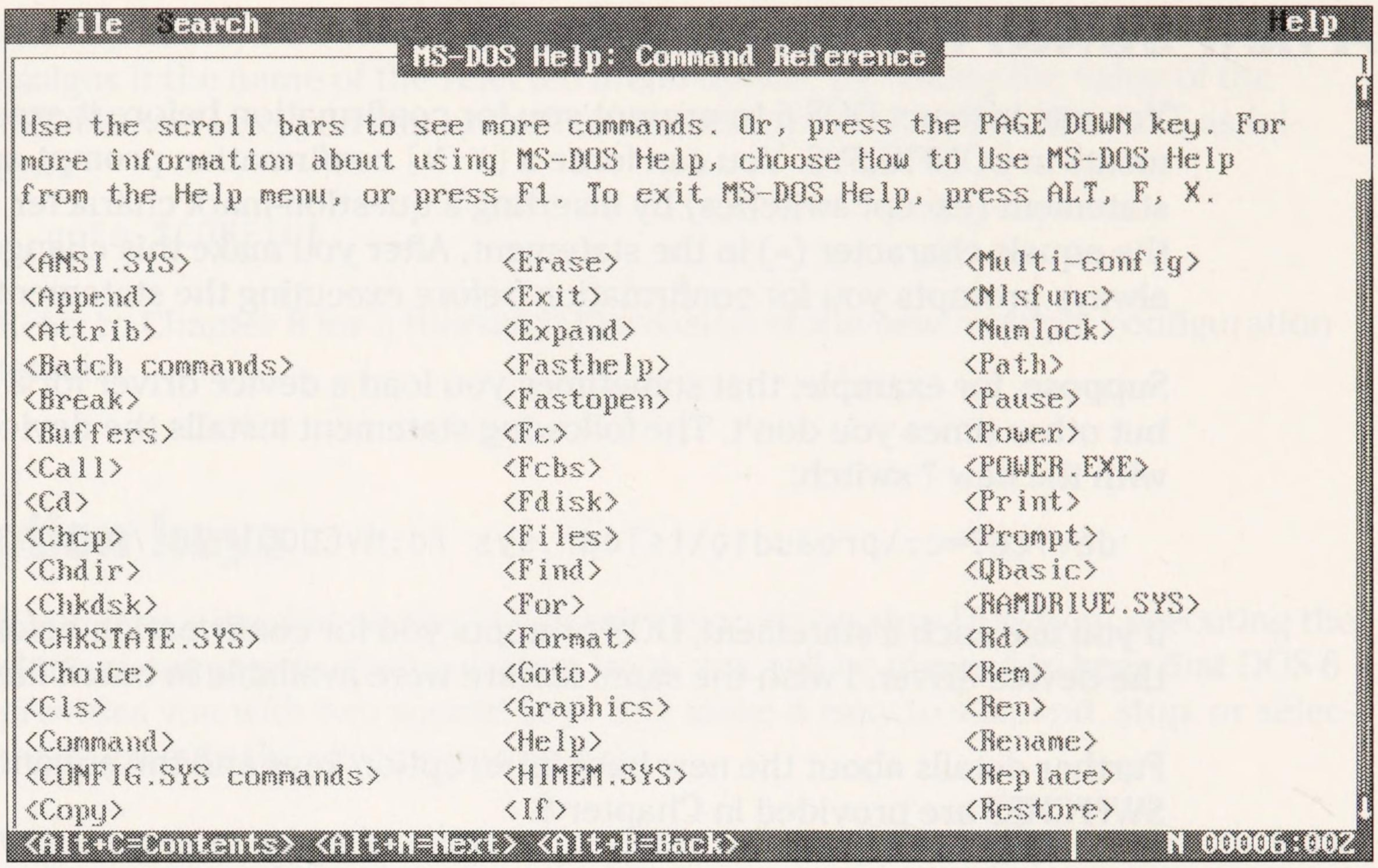
DOS 6 takes this idea further. It provides a full-blown hypertext help system that offers pages of useful information about almost every command and device driver. If you enter the command HELP with no switches, a full-screen menu is displayed, as illustrated in Figure 1-1.

You can request information about specific commands by clicking on the command name with the mouse or by tabbing to a command and pressing Enter. You can even search for all topics that contain a specific word or phrase. Most commands and device drivers have three sections of help: syntax, notes, and examples. Figure 1-2 shows part of the notes for the XCOPY command.

To exit the help system, press Alt-F, X.

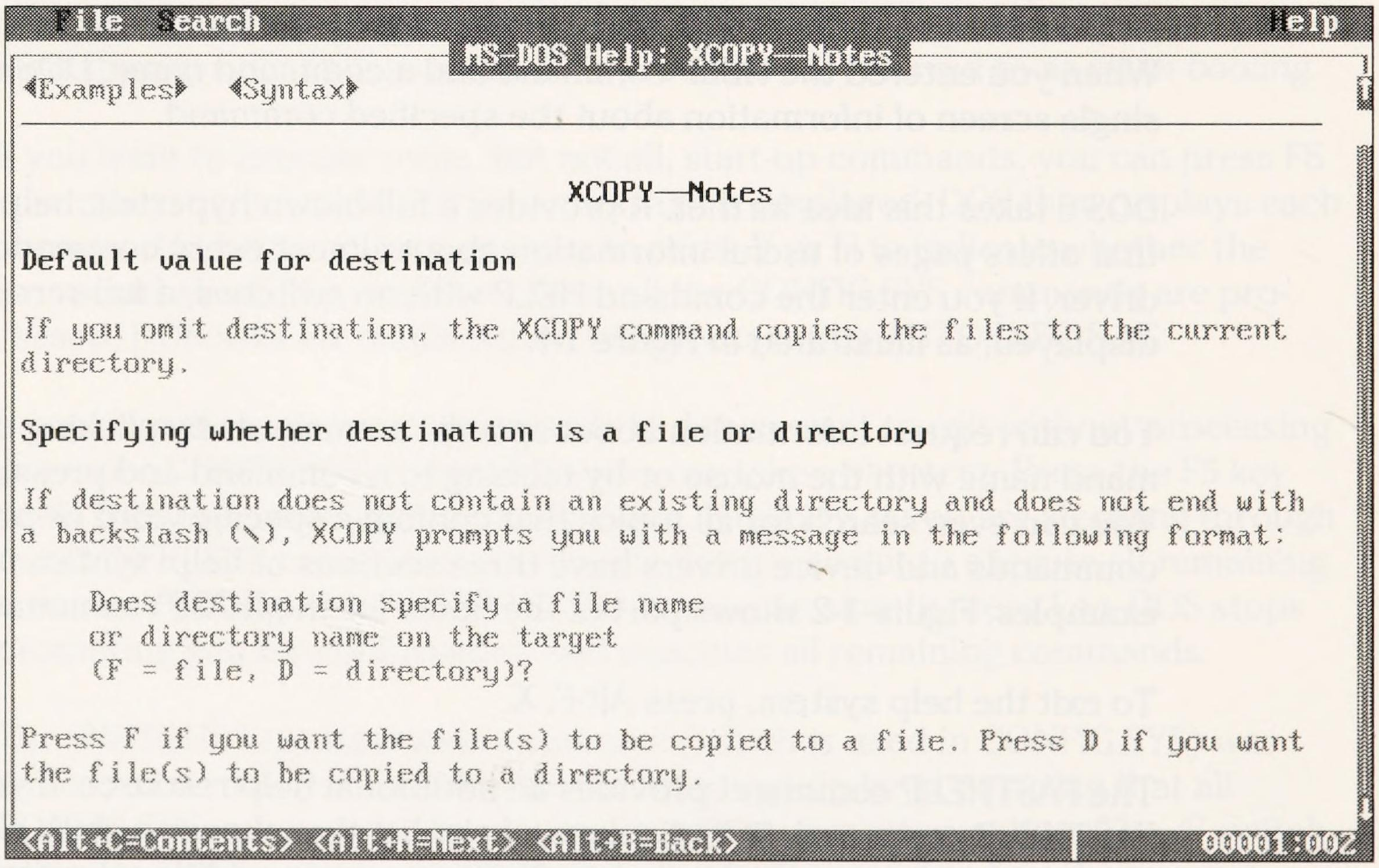
The FASTHELP command provides an additional help resource. If you enter the FASTHELP command, DOS displays a brief list that shows each DOS command along with the supported switches. You can access additional switch information by entering a specific command followed by the /? switch. For example, to display the switches supported by XCOPY, you enter the following command:

```
xcopy /?
```


File
Search
Help
MS-DOS Help: Command Reference
<Alt+C=Contents> <Alt+N=Next> <Alt+B=Back>
N 00006:002

```
Use the scroll bars to see more commands. Or, press the PAGE DOWN key. For
more information about using MS-DOS Help, choose How to Use MS-DOS Help
from the Help menu, or press F1. To exit MS-DOS Help, press ALT, F, X.

<ANSI.SYS>                   <Erase>                    <Multi-config>
<Append>                     <Exit>                     <Nlsfunc>
<Attrib>                     <Expand>                   <Numlock>
<Batch commands>             <Fasthelp>                 <Path>
<Break>                      <Fastopen>                 <Pause>
<Buffers>                    <Fc>                       <Power>
<Call>                       <Fcbs>                     <POWER.EXE>
<Cd>                         <Fdisk>                    <Print>
<Chcp>                       <Files>                    <Prompt>
<Chdir>                      <Find>                     <Qbasic>
<Chkdsk>                     <For>                      <RAMDRIVE.SYS>
<CHKSTATE.SYS>               <Format>                   <Rd>
<Choice>                     <Goto>                     <Rem>
<Cls>                        <Graphics>                 <Ren>
<Command>                    <Help>                     <Rename>
<CONFIG.SYS commands>        <HIMEM.SYS>                <Replace>
<Copy>                       <If>                       <Restore>
```

**Figure 1-1:** The Help menu.


File
Search
Help
MS-DOS Help: XCOPY—Notes
◄Examples►
◄Syntax►
<Alt+C=Contents> <Alt+N=Next> <Alt+B=Back>
00001:002

```
                           XCOPY—Notes

Default value for destination

If you omit destination, the XCOPY command copies the files to the current
directory.

Specifying whether destination is a file or directory

If destination does not contain an existing directory and does not end with
a backslash (\), XCOPY prompts you with a message in the following format:

    Does destination specify a file name
    or directory name on the target
    (F = file, D = directory)?

Press F if you want the file(s) to be copied to a file. Press D if you want
the file(s) to be copied to a directory.
```

**Figure 1-2:** An example of Help notes.

In general, the DOS 6 help system is excellent, but it presents a few minor irritations. The help information for some topics, for example, extends beyond one screenful. You can scroll down to read all the text, but when you jump from topic to topic, you encounter an annoying feature. Suppose that you're reading a lower part of the help screen for one topic and then jump to the help screen for a second topic. If you jump back to the first topic (by pressing Alt-B), DOS sends you back to the top line of that topic's help text, even though you were viewing a lower part of the text originally. Few other hypertext help systems, including that for Windows, suffer from this ailment.

Also, if you are viewing the main help index (or contents) and press a character, DOS jumps the cursor to the first command that begins with that letter. So far, so good. But if you immediately type a second character, DOS jumps to a command that begins with the new letter instead of homing in on a command that matches the first two characters. If you want to jump to the EDIT command, for instance, you might press the E key. DOS jumps to ECHO, the first command beginning with E. If you then press the D key, DOS jumps to the DATE command, the first command beginning with D. It should jump to EDIT, the only command beginning with the letters ED. Ah, well.

GEM

The main on-line help code is contained in the QBASIC.EXE file. Behind the scenes, HELP.COM calls QBASIC with the /QHELP switch (which must be uppercase). If you delete QBASIC.EXE, you won't be able to use the HELP command. Also, the help text is stored in the binary file HELP.HLP in the DOS directory. This file cannot be directly edited. However, if you have the Microsoft C Compiler version 6 or later, you can customize the help text. This technique is discussed further in the next chapter.

## Advanced Memory Management

Computer users want to get the most value from installed memory. With past versions of DOS, most *power* users resorted to third-party memory managers, such as QEMM, from Quarterdeck Office Systems, or 386MAX, from Qualitas, to make the best use of the 640K base memory. DOS 6, however, offers memory management capabilities similar to those provided by third-party memory managers. The new memory management tools come in the form of MEMMAKER and enhanced versions of EMM386.EXE, LOADHIGH, and DEVICEHIGH. These enhanced programs can squeeze more than 100K of extra upper memory out of a system — much more than was possible with DOS 5.

The bad news is that these tools only function on 386+ systems.

DISK

Although DOS seems to be leaving AT-class users behind, some terrific third-party utilities are available for these older systems. Bob's Shareware Collection disks include the excellent memory manager UMB_DRVR.SYS, from Christopher Blum. Among other things, UMB_DRVR.SYS can take advantage of the shadow

RAM on many of the 286 chip sets and can offer as much as 224K of upper memory on 286 systems. What's more, UMB_DRVR.SYS provides considerable advantages for 386 users, including the ability to load HIMEM.SYS high.

## Memory optimization tools

The art of memory optimization involves pushing as many programs and device drivers into upper memory as possible. The DEVICEHIGH and LOADHIGH commands are responsible for loading such software into upper memory. Both DEVICEHIGH and LOADHIGH are enhanced in DOS 6 with switches that enable you to control the precise amount and region of upper memory used for installation of a device driver or program.

You use the /L switch to identify the upper memory region where the software should be loaded. If the program needs more memory to install than it does to run, you can use the /S switch to shrink the memory allocation after the software is installed.

To help with memory analysis, the MEM command was enhanced so that it displays the amount of free UMBs when you use the /FREE switch. Also, MEM's /D switch shows which upper memory regions are occupied by each program.

Making optimum use of installed memory can be a complicated business. Even the order in which devices are loaded into upper memory can have a profound impact on the number of programs that can be loaded high. All these new switches are of little use if you can't determine which software should be loaded where and how much memory the programs need after they are installed.

Fortunately, DOS 6 has a very practical and effective solution to this problem. The program MEMMAKER analyzes your system, experiments a little, and then configures your system to make the best use of memory. All you have to do is configure your CONFIG.SYS and AUTOEXEC.BAT files so that your normal device drivers and programs are loaded — you don't need to worry about loading them high. If you then run MEMMAKER, DOS assesses your system and reconfigures it to make the most of installed memory.

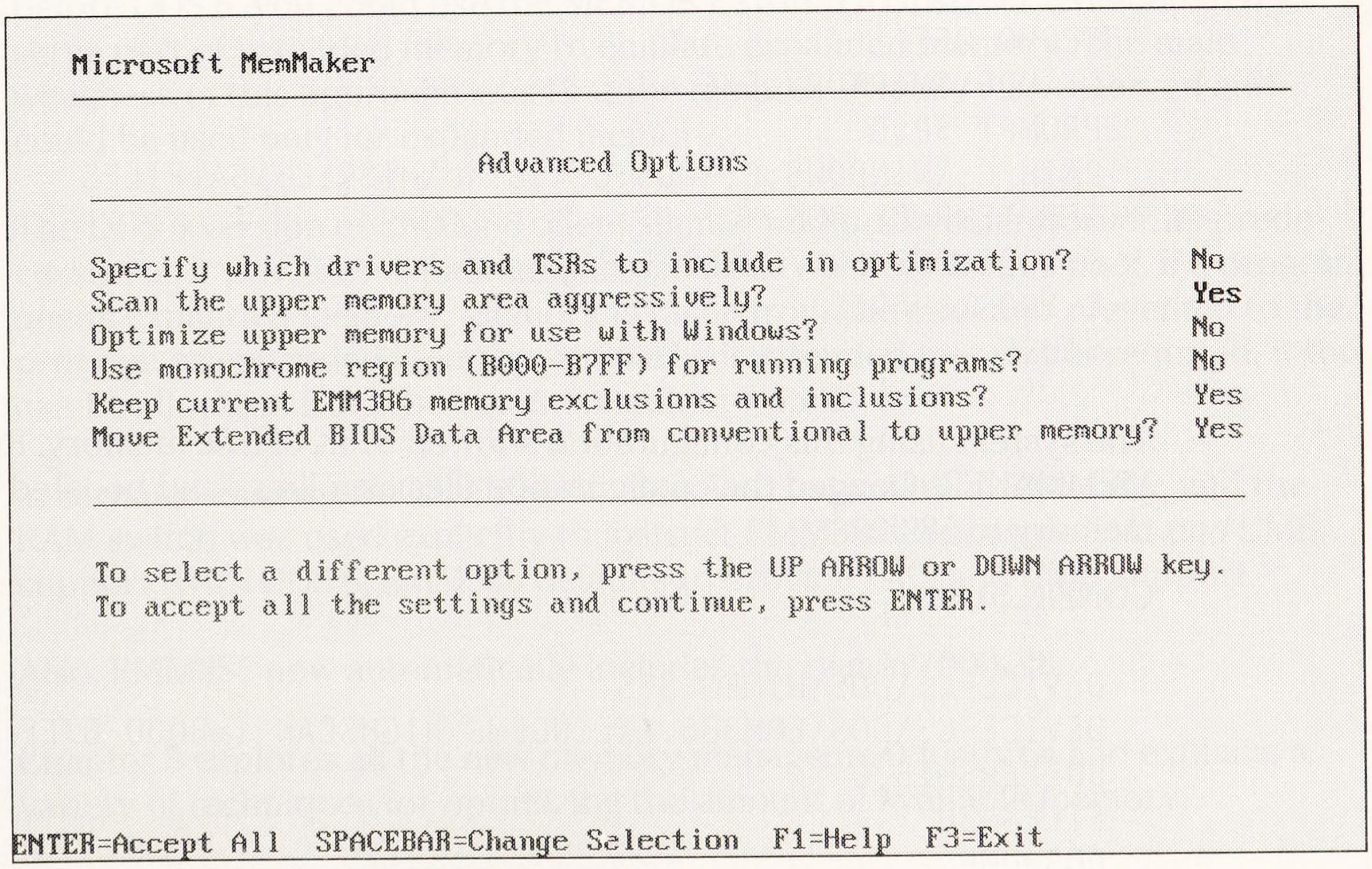
You can run MEMMAKER in Express Mode or Custom Mode. In Express Mode, MEMMAKER asks only one question before proceeding with the optimization: whether any programs need expanded memory support. If so, MEMMAKER automatically configures EMM386 to provide an EMS page frame in upper memory. Otherwise, that 64K of upper memory is used to create UMBs to load device drivers and programs high.

Selecting Custom Mode usually provides the highest degree of optimization and therefore increases the amount of available conventional memory. In Custom Mode, you can specify six advanced options (see Figure 1-3) that control many of the EMM386.EXE settings. One quick way to increase upper memory space on systems equipped with color displays is to set the Use monochrome region for running programs option to Yes.

After you make your selections, MEMMAKER moves ahead with the optimization procedure. Its primary objective is to determine how much memory device drivers and programs need. To perform this analysis, MEMMAKER uses two special utilities: CHKSTATE.SYS and SIZER.EXE.

MEMMAKER modifies CONFIG.SYS and adds a new line to the beginning of the file to load the CHKSTATE device driver. If the HIMEM.SYS, EMM386, and DOS=UMB statements are missing, MEMMAKER adds them. Each device statement is modified so that SIZER.EXE is used to load the device drivers and to observe them as they are loaded. AUTOEXEC.BAT is also modified so that SIZER.EXE can observe each program as it loads.

After the configuration files are modified, MEMMAKER reboots the system, and CHKSTATE.SYS and SIZER.EXE observe and gather statistics as the device drivers and programs are loaded. Using this data, MEMMAKER calculates which programs should be loaded into upper memory, in what order they should be



```
  Microsoft MemMaker

                              Advanced Options

   Specify which drivers and TSRs to include in optimization?    No
   Scan the upper memory area aggressively?                      Yes
   Optimize upper memory for use with Windows?                   No
   Use monochrome region (B000-B7FF) for running programs?       No
   Keep current EMM386 memory exclusions and inclusions?         Yes
   Move Extended BIOS Data Area from conventional to upper memory?  Yes

   To select a different option, press the UP ARROW or DOWN ARROW key.
   To accept all the settings and continue, press ENTER.

ENTER=Accept All  SPACEBAR=Change Selection  F1=Help  F3=Exit
```

**Figure 1-3:** Advanced MEMMAKER options.

loaded, and how much memory they need. In some extreme cases, MEMMAKER may compute more than 20,000 different configuration permutations before identifying the optimum one. Based on these computations, further edits are made to the configuration files, and the system is rebooted. Bingo, your system is optimized.

**GEM** You can use the file MEMMAKER.INF, located in the DOS directory, to control which device drivers and programs are not analyzed or executed by MEMMAKER. This file contains a list of programs and device drivers and can be easily customized with EDIT. Any program preceded by an asterisk is not modified by MEMMAKER but is executed. All other programs are completely ignored by MEMMAKER and are not run during the optimization process.

Listed next are the configuration files from a typical (not very well-configured) system:

CONFIG.SYS:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE
DEVICE=C:\DOS\DBLSPACE.SYS /MOVE
BUFFERS=15,0
FILES=30
DOS=HIGH
```

AUTOEXEC.BAT:

```
@ECHO OFF
C:\DOS\SMARTDRV.EXE
PROMPT $P$G
PATH C:\WINDOWS;C:\DOS;C:\;C:\U\MISC;C:\BATFILES
SET TEMP=C:\DOS
C:\DOS\DOSKEY /BUFSIZE=1024
C:\DOS\SHARE
MOUSE
```

The system using this configuration showed 515K of free memory. Running MEMMAKER changed the configuration files as follows and boosted the free memory to 589K:

CONFIG.SYS:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS HIGHSCAN I=B000-B7FF
BUFFERS=15,0
FILES=30
DOS=UMB
```

```
LASTDRIVE=I
FCBS=16,8
DEVICEHIGH /L:2,44432 =C:\DOS\DBLSPACE.SYS /MOVE
DOS=HIGH
```

AUTOEXEC.BAT:

```
@ECHO OFF
LH /L:0 C:\DOS\SMARTDRV.EXE
PROMPT $P$G
PATH C:\WINDOWS;C:\DOS;C:\;C:\U\MISC;C:\BATFILES
SET TEMP=C:\DOS
LH /L:1,6400 C:\DOS\DOSKEY /BUFSIZE=1024
LH /L:1,13984 C:\DOS\SHARE
LH /L:2,56928 MOUSE
```

Unfortunately, MEMMAKER is not adept at handling a CONFIG.SYS file with multiple configurations. The only practical solution is to run MEMMAKER several times, changing CONFIG.SYS and AUTOEXEC.BAT each time to reflect a different configuration option. After optimizing each configuration separately, you can then rebuild a consolidated CONFIG.SYS and AUTOEXEC.BAT. Ugh.

## EMM386 enhancements

Before DOS 6, you could use the MEMORY switch to instruct EMM386 to reserve part of extended memory to emulate expanded memory. The main weakness of this approach was that the memory literally was reserved and could be used only for expanded memory.

The DOS 6 version of EMM386 offers significant improvements over its predecessor, as it provides expanded and extended memory from a *pool*. If a program needs expanded memory, memory is taken from the pool and presented to the program as expanded memory. Similarly, if extended memory is required, it too can be allocated from the common memory pool.

In earlier versions of DOS, EMM386 defaulted to providing EMS only, and the RAM switch was used explicitly to instruct EMM386 to provide EMS *and* UMB support. With DOS 6, EMM386 defaults to provide EMS and UMB support.

Also, EMM386 now automatically includes the region c000-efff.

Chapter 5 explores all the new memory management features and explains a variety of techniques for optimizing the amount of available memory.

## Virus Protection

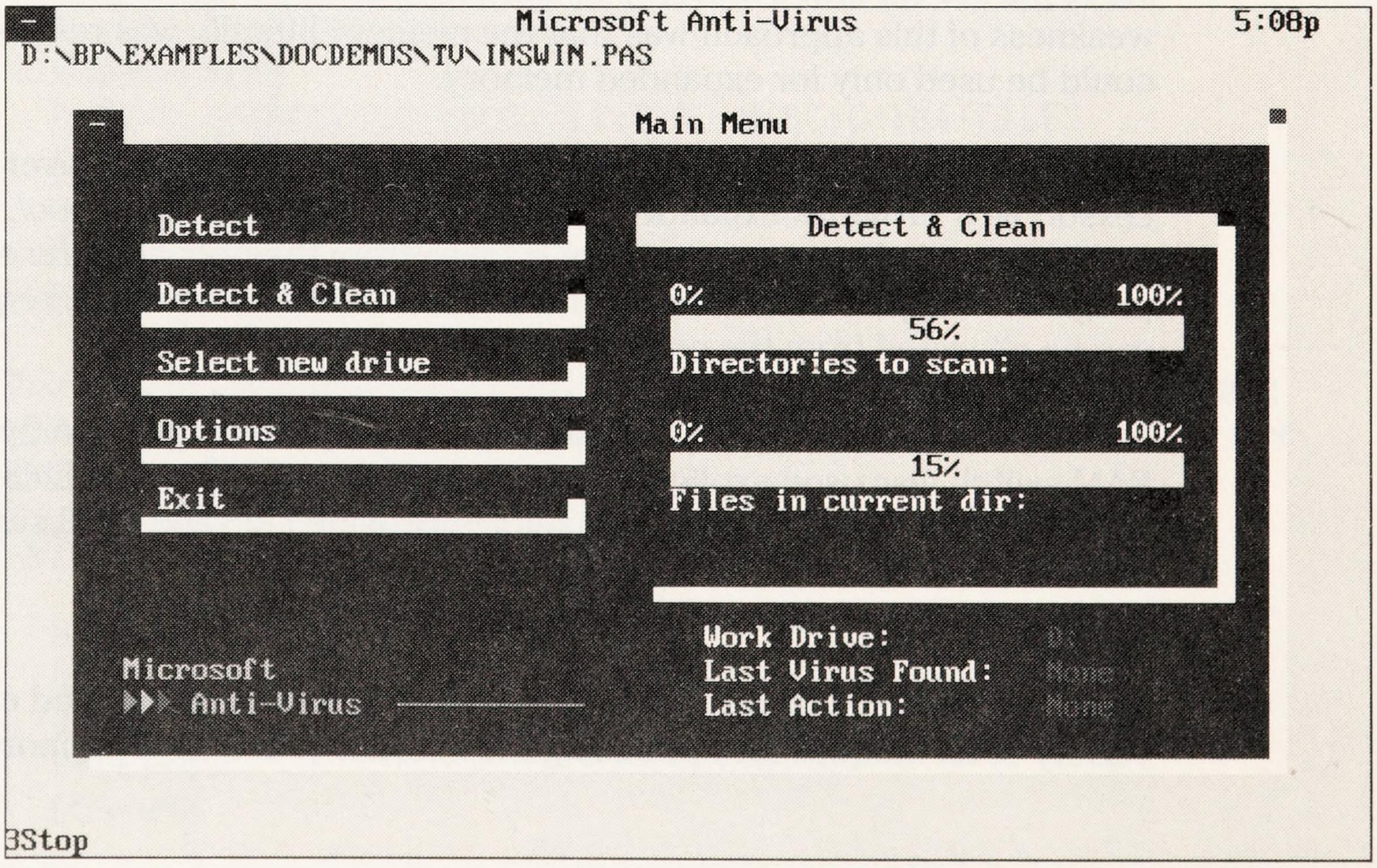
A software virus is a program that is designed to propagate from system to system undetected. After some trigger event, such as booting the system on or after a specific date, many virus programs try to destroy valuable data by damaging the FAT, formatting the disk, or randomly writing to disk sectors. Virus programs are unwanted and harmful. Although it is true that far more data is destroyed by user error, every prudent user should take measures to protect against virus infection.

For the first time, DOS includes anti-virus tools. The MSAV program can search for and remove viruses, and the memory-resident VSAFE program can guard against infection and protect the disk areas commonly attacked by viruses.

A Windows version of MSAV, called MWAV, is discussed in a later section.

MSAV is a full-screen program that provides two options. You can just search for viruses, or you can perform the search and eradicate any viruses that are discovered (see Figure 1-4). These processes are referred to as *scanning* and *cleaning,* respectively.

Most known viruses leave a specific pattern of bytes, known as *virus signatures,* in an infected file. MSAV searches for these signatures in all files on the selected drive(s). New virus strains are discovered every month, so MSAV must be



Microsoft Anti-Virus

D:\BP\EXAMPLES\DOCDEMOS\TV\INSWIN.PAS

Directories to scan:

Files in current dir:

Work Drive: D:

Last Virus Found: None

Last Action: None

**Figure 1-4:** The full-screen MSAV program.

updated on a regular basis to look for these signatures. You can update the DOS virus programs by downloading the updated signature files from a dedicated bulletin board system. Further details are explained in Chapter 3.

MSAV also offers an optional generic virus-identification method that uses *checksums* to identify suspicious changes in a program file. A checksum is a numerical value derived by applying a special algorithm to the individual bytes in a file. When a file is changed, so is the checksum. By comparing a file's current computed checksum to a checksum created previously, you can ascertain whether the file has been changed.

MSAV is designed to scan an entire drive and cannot be configured to scan an individual directory. Bob's Shareware Collection disks include SCAN, from McAfee Associates. SCAN is a very flexible and fast virus-detection utility and is part of the renowned anti-virus tools from McAfee Associates. Refer to Part V for further information.

VSAFE is a memory-resident program that monitors program and disk activity. If some program tries to make a suspicious change to the disk (such as modifying the boot sector or an EXE file), VSAFE displays a dialog box to alert you and prompts you for confirmation before allowing the change to occur. VSAFE does not check for specific brands of viruses. It protects the areas usually infected by viruses and so provides generic protection against all viruses.

## Heavy-Duty Disk Utilities

Until recently, DOS provided third-rate tools for managing data stored on hard disks. BACKUP and RESTORE were jokes, the disk cache was one of the slowest on the market, and programs such as RECOVER did more harm than good.

Many software developers built large corporations by creating products that filled the gap in the DOS hard-disk arsenal. At long last, Microsoft has acknowledged that the operating system needs to provide real disk utilities and has introduced some industrial-strength disk utilities in DOS 6. In fact, the UNDELETE and BACKUP tools (as well as the anti-virus tools discussed earlier) were licensed from some of the major disk-utility companies. If you can't beat 'em, join 'em.

### DoubleSpace disk compression

One of the most eagerly anticipated features of DOS 6 was the all-new DoubleSpace disk compression facility. Like SuperStor, from Addstor Inc., and Stacker, from Stac Electronics, DoubleSpace uses space on an existing drive to create a new compressed drive. For example, DoubleSpace can take 20MB of free space on a drive and create a new drive that can hold about 40MB of files. Similarly, it can compress a 100MB C drive to provide storage for approximately 200MB of files.

A DoubleSpace drive actually is a large hidden file that is stored in the root directory of a physical drive. The DoubleSpace device driver DBLSPACE.BIN tricks DOS into thinking that there is an additional hard drive, in much the same way that RAMDRIVE.SYS makes DOS believe an area of memory is a disk.

Whenever a program (including DOS) tries to save data to a compressed drive, DoubleSpace intercepts the request, compresses the data, and then saves the compressed data in the hidden file. When a program tries to read a file from the compressed drive, DoubleSpace reads the data from the hidden file, decompresses the data, and then passes it on to the program. The programs are oblivious to the data compression tasks taking place behind the scenes.

You might think that all this behind-the-scenes compression and decompression would significantly slow down your system, but that isn't usually true. One of the slowest procedures in a PC is the physical reading and writing of data to the surface of the disk. In many cases, the delay incurred while DOS compresses and decompresses data is offset by the time that is saved because 50 percent less data must be physically accessed on the disk surface. Although the exact impact of disk compression varies from system to system, a broad average seems to be that systems operate approximately 5 to 10 percent slower when accessing a compressed drive.

## DBLSPACE.BIN

DoubleSpace is arguably the easiest to use of all the disk compression tools, especially when you want to compress your boot drive (for example, drive C). Because DOS accesses the hidden files MSDOS.SYS and IO.SYS during system boot, DOS must boot from an uncompressed drive. After all, DOS wouldn't be able to read these system files if they were compressed.

Most third-party tools maintain two copies of CONFIG.SYS (and sometimes AUTOEXEC.BAT). One copy of CONFIG.SYS is on the uncompressed drive and contains the statement to load the disk compression device driver. As soon as the device driver is loaded, the compressed drive is activated and swapped with the boot drive. The physical boot drive is assigned a new letter (for example, F), and the compressed drive is assigned the letter C.

So far, so good. The problem is that most software (not to mention most users) expects drive C and not drive F (or whatever) to contain the CONFIG.SYS and

AUTOEXEC.BAT files. Two copies of CONFIG.SYS are maintained by the disk compression software. One copy is on the compressed drive (drive C), and another is on the boot drive. Whenever the system boots, the disk compression utility compares the two copies and keeps them synchronized. Sometimes, this requires a second boot during start-up.

DoubleSpace is smarter. There is no device driver entry in CONFIG.SYS for DBLSPACE.BIN, the disk compression device driver. Before even looking at CONFIG.SYS, IO.SYS searches the root directory of the boot drive, looking for the device driver file. If the file is found, it is automatically loaded. Only after the device driver is loaded and the physical boot drive is swapped with the compressed drive does IO.SYS look for CONFIG.SYS. The result is that the real CONFIG.SYS and AUTOEXEC.BAT files reside on the compressed drive.

GEM

Because DBLSPACE.BIN is loaded into memory before CONFIG.SYS is processed, it must be loaded into conventional memory. The upper memory area can be used only after HIMEM.SYS and EMM386 are installed. DBLSPACE.BIN is initially loaded at the top of memory, adjacent to the memory area occupied by IO.SYS.

DOS provides another DoubleSpace device driver, called DBLSPACE.SYS, whose purpose is to transfer DBLSPACE.BIN from base memory to upper memory. This device driver is loaded from CONFIG.SYS in the normal manner, using a DEVICE or DEVICEHIGH statement. If CONFIG.SYS contains a statement loading DBLSPACE.SYS with DEVICE, DBLSPACE.BIN is moved to the bottom of base memory. If DBLSPACE.SYS is loaded with DEVICEHIGH, DBLSPACE.BIN is moved to upper memory.

## The DBLSPACE command

The DBLSPACE command is used to manage compressed drives. The first time you execute DBLSPACE, it guides you through the process of creating a compressed drive. Like MEMMAKER, DBLSPACE offers an Express Mode and a Custom Mode, and both are very easy to use. In Express Mode, DBLSPACE compresses the boot drive along with all the files on it. In Custom Mode, you can choose which drive is to be compressed, indicate whether existing files should be compressed or left on an uncompressed drive, and control the size and drive letter of the compressed drive.

After you create at least one compressed drive on your system, you can use DBLSPACE to maintain and manage the compressed drives. DBLSPACE can be executed as a command-line utility or as a full-screen utility. For example, you can use the DBLSPACE /LIST command to see a summary of all the drives on your system. The following is a sample of such a summary:

```
Drive Type                    Total Free  Total Size  CVF Filename
----------------------------------------------------------------------

  A   Removable drive         No disk in drive
  B   Floppy drive               0.04 MB     0.69 MB
  C   Local hard drive           7.79 MB   201.97 MB
  D   Local hard drive          29.87 MB   165.11 MB
  E   Local hard drive          14.82 MB   159.32 MB
  F   Available for DoubleSpace
  G   Available for DoubleSpace
  H   Available for DoubleSpace
  I   Compressed hard drive     29.31 MB    37.00 MB E:\DBLSPACE.001
  J   Compressed hard drive     43.65 MB    43.65 MB D:\DBLSPACE.001
```

Notice that in this example, the compressed drives I and J are actually hidden files called DBLSPACE.001 on drives E and D, respectively.

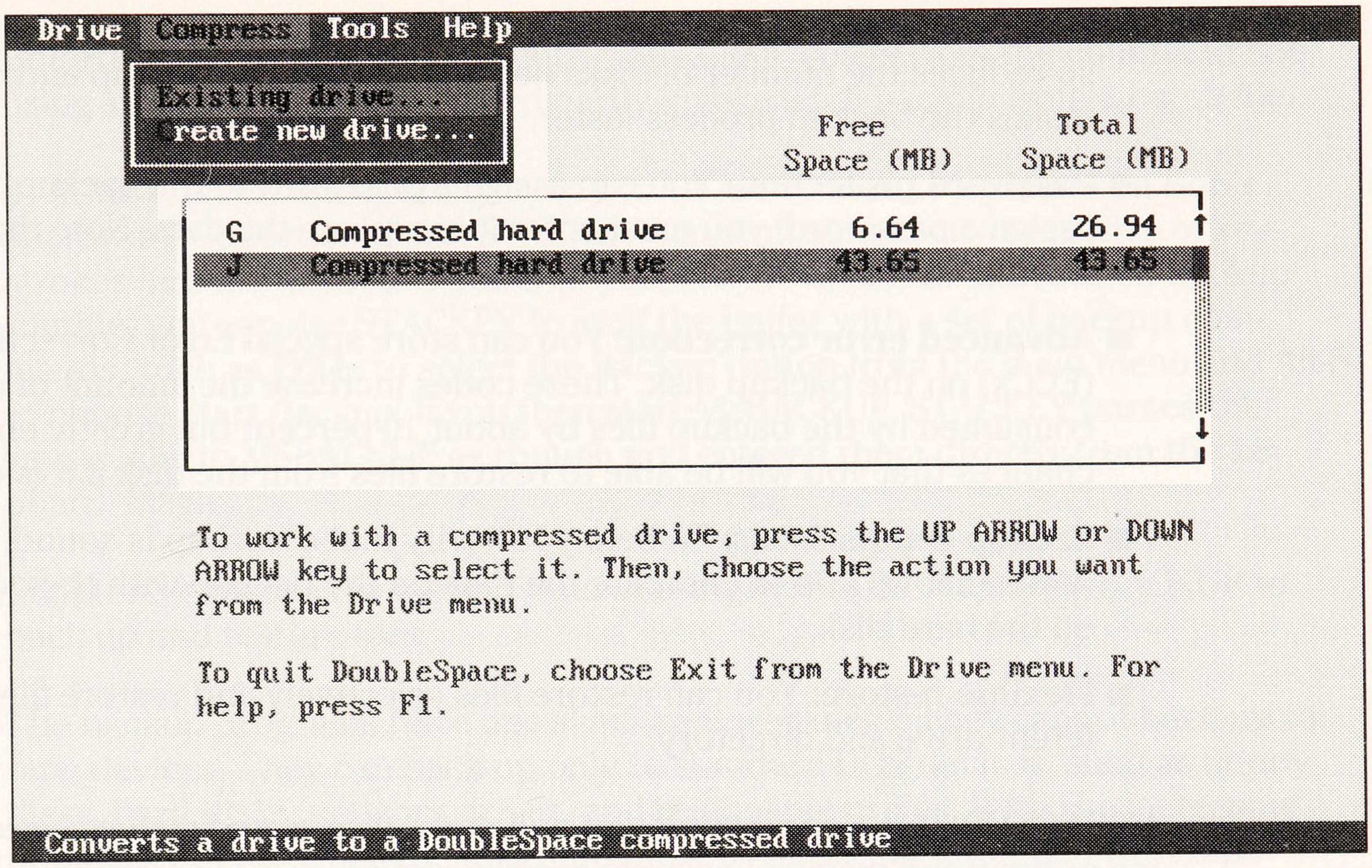
DoubleSpace allocates space on a sector-by-sector basis, whereas DOS allocates space by clusters. Like standard hard drives, a compressed drive has a boot sector, FAT, and root directory. It also maintains a sector allocation table that maps standard FAT clusters to DoubleSpace sectors.

The hard disk maintenance tools, such as CHKDSK and DEFRAG, are enhanced in DOS 6 so they can perform work on compressed drives. If you execute such utilities on a compressed drive, DOS automatically calls the DoubleSpace versions after the versions that work on the host drive are done.

You can execute DBLSPACE with no switches to launch the full-screen application, as shown in Figure 1-5. From the DoubleSpace menu, you can perform maintenance tasks, such as running CHKDSK or defragmenting files; create new compressed drives; and delete or change the size of existing compressed drives.

Chapter 4 explores many different ways to take full advantage of the new DoubleSpace tools.

DoubleSpace does support compression of data on diskettes but only in a clumsy way. Even if you don't have any compressed drives on the hard disk, you must boot your system so that the DoubleSpace device driver is installed — that is, with DBLSPACE.BIN in the root directory of the boot drive — just to have compressed data on floppies. This occupies more than 40K of precious RAM. Even more cumbersome is the way you have to mount (with DBLSPACE) each and every compressed diskette before you can access the data on it.



|   |   | Free Space (MB) | Total Space (MB) |
|---|---|---|---|
| G | Compressed hard drive | 6.64 | 26.94 |
| J | Compressed hard drive | 43.65 | 43.65 |

To work with a compressed drive, press the UP ARROW or DOWN ARROW key to select it. Then, choose the action you want from the Drive menu.

To quit DoubleSpace, choose Exit from the Drive menu. For help, press F1.

Converts a drive to a DoubleSpace compressed drive

**Figure 1-5:** The full-screen DBLSPACE utility.

## A real backup program

For a long time, the DOS utilities BACKUP and RESTORE were totally outclassed by third-party backup utilities. Experienced DOS users were forced to use non-DOS backup programs because the DOS offerings were so inadequate. Finally, with the introduction of version 6, DOS users are given a practical and powerful backup program.

The long-standing DOS program BACKUP is replaced by two new backup programs: MSBACKUP for DOS and MWBACKUP for Windows. Both programs are licensed from Symantec, the makers of Norton Backup.

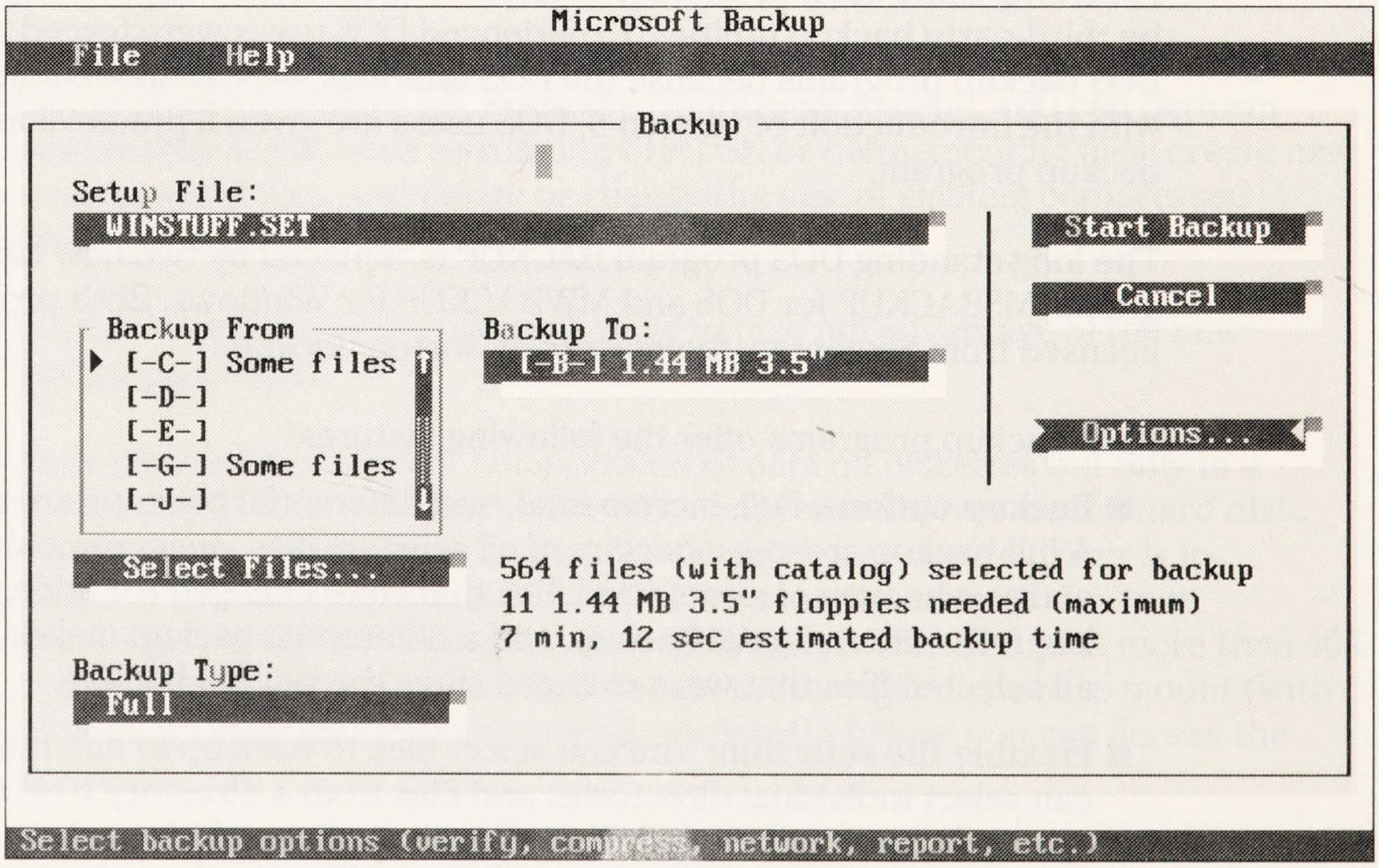
Both backup programs offer the following features:

- **Backup options:** Full, incremental, and differential backups are supported. A full backup creates a backup of all selected files, an incremental backup makes a backup of all selected files that were changed (or added) since the last full or incremental backup, and a differential backup makes a backup of all selected files that were changed since the last full backup.
- **Flexible file selection:** You can select files to back up in different ways. You can select individual directories and files from a directory tree, or you can specify include and exclude lists that use paths and filemasks to identify the file selection. You also can use a combination of both methods.

- **Data compression:** Data can be compressed before it is saved to disk. Doing so reduces the number of disks needed to store the backup and often makes the backup process faster.
- **Password protection:** You can assign a password to the backup. After you assign a password, you must enter it to restore the data. Note that the password *is* case sensitive.
- **Advanced error correction:** You can store special Error Correction Codes (ECCs) on the backup disk. These codes increase the amount of disk space consumed by the backup files by about 10 percent but greatly enhance the chances that you will be able to restore files from the disk if it is damaged.
- **Backup comparisons:** To help you verify that a backup is sound, a compare option allows you to compare the data on the backup with the original data on the hard disk.
- **Flexible restore:** You can restore files selectively and restore files to a different drive and directory.

MSBACKUP uses a combination of pull-down menus and dialog boxes for selecting options and making backups (see Figure 1-6).

The Windows version of the backup program, MWBACKUP, is discussed later, in the section titled “Windows-Hosted Utilities.”



Setup File: WINSTUFF.SET

Backup From:
[-C-] Some files
[-D-]
[-E-]
[-G-] Some files
[-J-]

Backup To: [-B-] 1.44 MB 3.5"

564 files (with catalog) selected for backup
11 1.44 MB 3.5" floppies needed (maximum)
7 min, 12 sec estimated backup time

Backup Type: Full

Select backup options (verify, compress, network, report, etc.)

Figure 1-6: The MSBACKUP dialog box.

Unfortunately, you cannot execute MSBACKUP solely from the command line using switches, so you can't ordinarily invoke a full backup from a batch file.

STACKEY, an excellent utility from CtrlAlt Associates, is included on Bob's Shareware Collection disks. STACKEY can stuff almost any sequence of keystrokes into the keyboard buffer. If you want to create an automated backup routine, you can use STACKEY to stuff the buffer with a set of backup commands, such as Enter to select the Backup option from the main menu and Alt-S to choose Start Backup. If you then start MSBACKUP, STACKEY passes the keystrokes to MSBACKUP as though you entered them directly from the keyboard. Magic.

Chapter 3 explains the ins and outs of using MSBACKUP and MWBACKUP to back up and restore files.

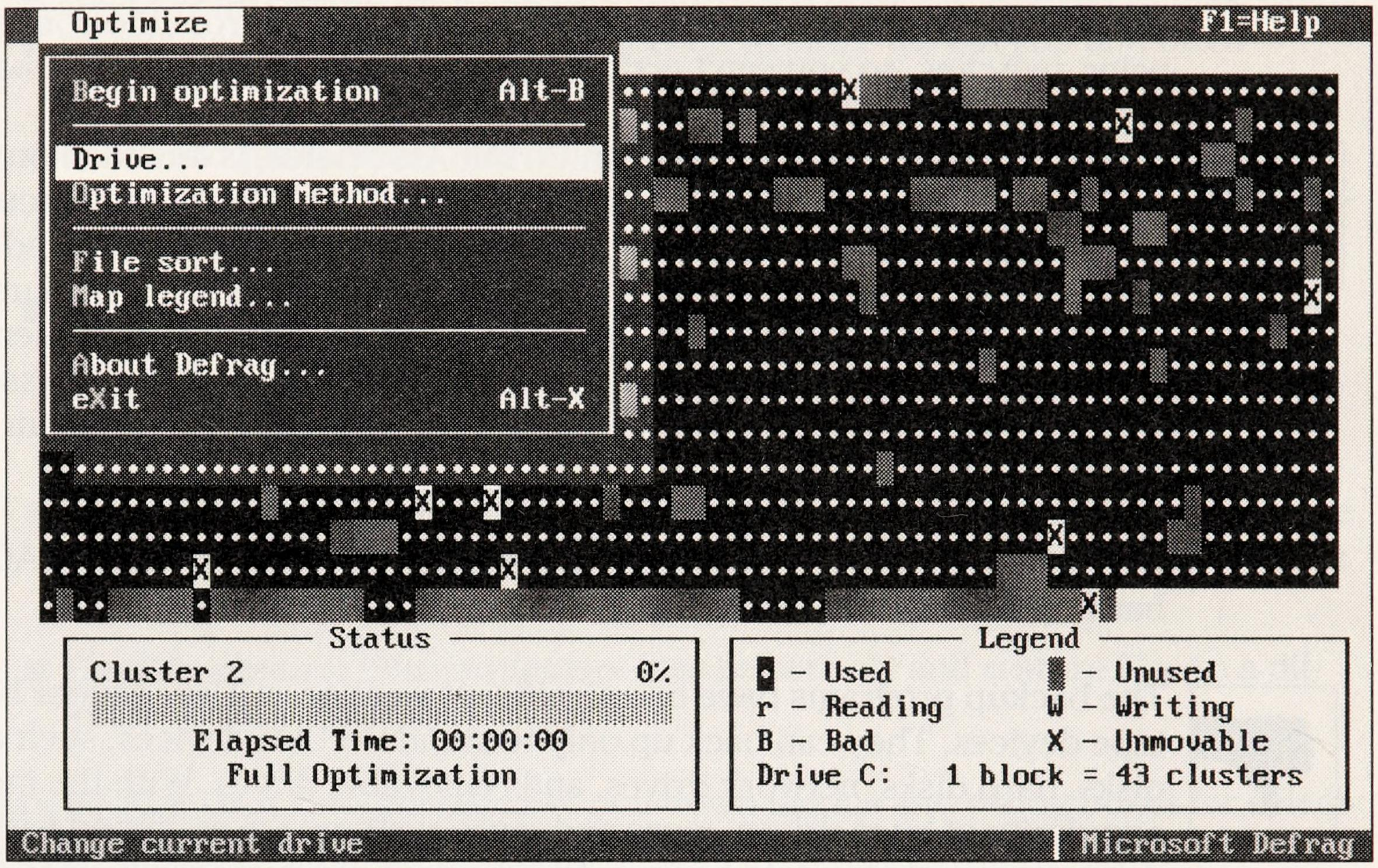
The backup programs have one significant weakness: They cannot back up to tape devices. They can back up only to standard DOS devices, such as floppy disks, hard disks, network drives, and removable drives. With the increasing popularity of huge drives, more and more systems are equipped with tape drives expressly for data backup. DOS backup products really need to support tape drives. No sensible third-party vendor would dream of releasing a new backup program that didn't support tape drives.

## Disk defragmentation

A file is said to be *fragmented* when it is not stored in contiguous clusters. Although DOS can manage and handle fragmented files reliably, you should always strive to minimize file fragmentation. It takes longer to access fragmented files than contiguous files because the disk read-write heads must move around the disk surface accessing the different locations where various pieces of the file are stored. What's more, the data recovery tools UNDELETE and UNFORMAT are much more likely to be successful if the files they are trying to recover are not fragmented.

DOS 6 provides a defragmentation tool called DEFRAG, which is licensed from Symantec. DEFRAG can defragment files on any drive by rearranging the clusters on the disk. If you execute DEFRAG with no switches, you can use the program interactively, selecting commands and options from menus. Figure 1-7 shows the main DEFRAG screen, which provides a graphical representation of the disk fragmentation.

By using switches, you can control the entire defragmentation process from the command line. You can specify the optimization method, choose the sort order for directories, and even instruct DEFRAG to reboot the system when it finishes.



Status
Cluster 2 0%
Elapsed Time: 00:00:00
Full Optimization

Legend
▪ - Used
▒ - Unused
r - Reading
W - Writing
B - Bad
X - Unmovable
Drive C: 1 block = 43 clusters

Change current drive | Microsoft Defrag

**Figure 1-7:** A DEFRAG disk map showing the locations of free space on a fragmented disk.

DEFRAG uses only conventional memory to defragment a disk. Because every directory and file is loaded into memory, DEFRAG doesn't have enough memory to handle large drives with many files and directories. As a rule of thumb, drives less than 400MB in size can be defragmented.

Disk OrGanizer™ (DOG for short), from Soft GAM's Software, is included on Bob's Shareware Collection disks. DOG™ provides all the functionality of the DOS DEFRAG command and then some. For example, DOG™ uses EMS and so can defragment disks with as many as 32,000 files.

## New undeletion tools

Microsoft licensed UNDELETE and MWUNDEL from Central Point Software to provide the primary DOS data-recovery tools. (The UNFORMAT and MIRROR commands also are licensed from Central Point Software.)

UNDELETE is a DOS command-line utility that enables you to protect against accidental file deletion and to recover files after they have been deleted. When loaded as a TSR, UNDELETE consumes approximately 14K of RAM and monitors all file-deletion activity. Whenever a file is deleted, UNDELETE saves information that enables you to recover the file. UNDELETE offers the following two methods of protecting files:

- **Delete Sentry:** Copies of deleted files are stored in a hidden directory named \SENTRY. This method provides the most secure form of protection because the file can be undeleted just by moving it from the deleted directory back to the original directory.
- **Delete Tracker:** The name of the deleted file, along with a list of all the clusters where the file's data was stored, is kept in a hidden file named PCTRACKR.DEL. This method doesn't offer as much protection as the Delete Sentry method because the clusters still can be overwritten by other files. However, it consumes much less disk space.

You can use the UNDELETE and MWUNDEL commands to recover a deleted file. They attempt to recover a deleted file even if the sentry or tracker defense mechanisms are not loaded. If you have Windows installed, use MWUNDEL — it is much more visual and easier to use than UNDELETE.

Refer to Chapter 3 for a thorough explanation of how to use UNDELETE and MWUNDEL to protect and recover files, along with other techniques for guarding against data loss.

### Improved cache performance

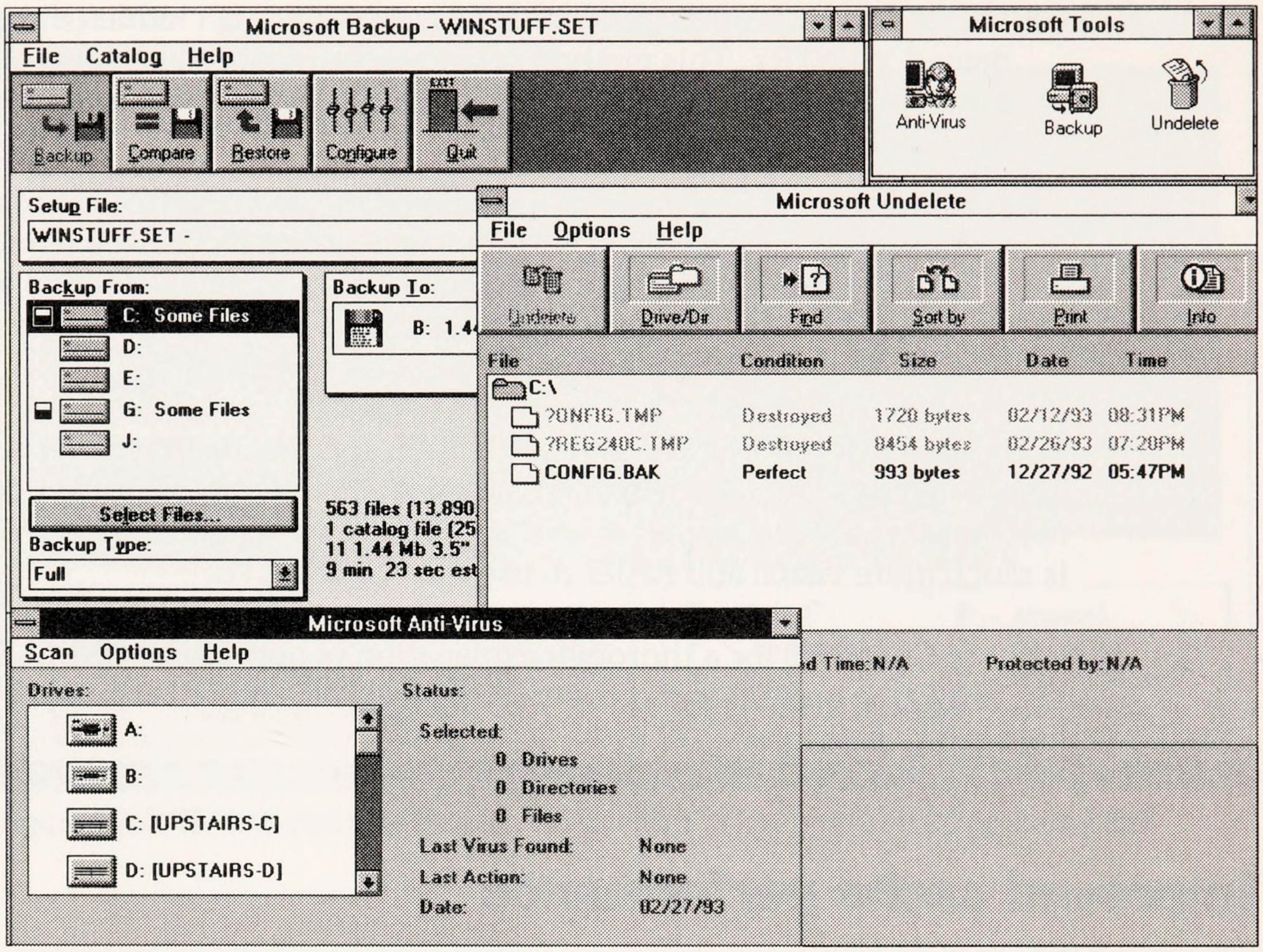
One of the most effective ways to improve the performance of a system is to use a disk cache. DOS 6 includes version 4.1 of the Microsoft disk cache SMARTDRV. Not all disk caches are alike, and in the past, SMARTDRV tended to be one of the poorer performers. To give SMARTDRV a performance boost, Microsoft added support for *write buffering*. When a program saves data to disk, SMARTDRV stores the data in a memory cache. The program, thinking that the file has been saved, can then proceed with other operations while the disk cache software saves the data to disk in the background. There is, however, an increased chance for data loss if the system is rebooted (deliberately or by a power failure or system crash) before the cache has saved the data to disk.

Thanks to write buffering and other enhancements, SMARTDRV is now a viable disk cache for the power user.

SMARTDRV cannot cache CD-ROM drives. If you want to cache a CD-ROM drive, you must use a different cache program.

## Windows-Hosted Utilities

More and more PCs are running Windows. For the first time in the history of DOS, some Windows-hosted utilities are included with the operating system. The three main utilities are MWAV, MWBACKUP, and MWUNDEL, which provide anti-virus, file backup, and file-undeletion facilities, respectively. All three


Microsoft Backup - WINSTUFF.SET
File
Catalog
Help
Backup
Compare
Restore
Configure
Quit
Microsoft Tools
Anti-Virus
Backup
Undelete
Microsoft Undelete
File
Options
Help
Undelete
Drive/Dir
Find
Sort by
Print
Info
Select Files...
Microsoft Anti-Virus
Scan
Options
Help

Setup File: WINSTUFF.SET -

Backup From:

```
C:  Some Files
D:
E:
G:  Some Files
J:
```

Backup To: B: 1.4

Backup Type: Full

```
563 files (13,890
1 catalog file (25
11 1.44 Mb 3.5"
9 min  23 sec est
```

```
File            Condition   Size         Date      Time
C:\
 ?ONFIG.TMP     Destroyed   1720 bytes   02/12/93  08:31PM
 ?REG240C.TMP   Destroyed   8454 bytes   02/26/93  07:20PM
 CONFIG.BAK     Perfect     993 bytes    12/27/92  05:47PM
```

```
d Time: N/A        Protected by: N/A
```

Drives:

```
A:
B:
C: [UPSTAIRS-C]
D: [UPSTAIRS-D]
```

```
Status:
  Selected:
      0  Drives
      0  Directories
      0  Files
  Last Virus Found:   None
  Last Action:        None
  Date:               02/27/93
```

**Figure 1-8:** The Windows tools providing anti-virus, backup, and file-undeletion support.

programs have text-mode counterparts, and when you install DOS 6, you can elect to install only the Windows version of the programs, only the DOS version, or both. In general, the DOS and Windows programs offer similar features, and the choice is purely a cosmetic one. Most Windows users, however, opt to use the Windows-hosted versions.

Figure 1-8 shows the three Windows applications and the Microsoft Tools group created by SETUP.

A small Windows utility, SMARTMON, also is included with DOS 6. SMARTMON visually displays the effectiveness of the SMARTDRV disk cache and enables you to alter the cache mode of any drive.

## Laptop Tools

DOS 6 provides the InterLink programs INTERLNK.EXE and INTERSVR.EXE to connect two PCs together for transferring files or sharing a printer. One computer runs INTERLNK as a device driver, and the other runs the program INTERSVR. Although designed primarily for laptop users, these programs can link any two computers that are wired together by a bidirectional parallel cable, a three-wire serial cable, or a seven-wire null modem serial cable.

You can't use the host computer to perform any other tasks while it's running INTERSVR (even if INTERSVR is executed from within Windows), but the computer that loaded INTERLNK can access the drives and printers on the host. The drives and printers on the host are assigned to additional drive letters and LPT ports on the computer running INTERLNK.

Data can be transferred to and from the computers using standard DOS commands. Programs treat the remote drives as additional local drives.

Refer to Appendix D for a detailed explanation of how to link two computers using InterLink. The appendix also explains how to connect a DOS 6 computer to a network using Microsoft's Workgroup Connection software.

InterLink is considerably slower than many similar third-party offerings. It is fine for low-volume file transfers and print jobs but is very sluggish when large data transfers are involved.

In addition to the InterLink programs, DOS 6 provides POWER.EXE, which is both a device driver and an executable program. You can use POWER to conserve battery power on battery-powered computers that support the industry-standard Advanced Power Management (APM) specification.

## Other Goodies

Other commands introduced in DOS 6 include the following:

- **CHOICE:** By using the CHOICE command in a batch file, you can prompt a user to press a key to select an option. CHOICE sets the ERRORLEVEL to indicate which key was pressed.

  Chapter 10 provides a whole suite of examples explaining how to use CHOICE in batch files. CHOICE can be enhanced with DEBUG to accept keys like Esc and Enter. Chapter 11 explains how to customize CHOICE with DEBUG.

- **DELTREE:** This powerful and potentially destructive command removes a directory, all of its subdirectories, and all files contained in the directories.

  Even if the read-only attribute is set on a file or directory, DELTREE will cheerfully blow it away without even asking for confirmation. Always use this command with caution.

- **MOVE:** You can use this command to move files from one directory to another.

- **MSD:** MSD is short for Microsoft System Diagnostics. This excellent program was introduced in Windows 3.1 and was so popular it was added to the DOS command set. It provides detailed information about your system's hardware and software configuration.

## What's Gone?

Some DOS commands have been atrophying for a number of years, and Microsoft finally decided to remove them. These commands are still available on the DOS 6 Supplemental disk, which you can obtain directly from Microsoft. Programs dropped from the main DOS package include the following:

ASSIGN.COM
BACKUP.EXE
COMP.EXE
CV.COM
EDLIN.EXE
EXE2BIN.EXE
GRAFTABL.COM
JOIN.EXE
MIRROR.COM
MSHERC.COM
PRINTFIX.COM
PRINTER.SYS

NOTE

If you are upgrading from an earlier version of DOS, SETUP does not remove old copies of these programs. After you upgrade to DOS 6, the old versions of these commands remain in the DOS directory. If you try to use some of these old programs (such as ASSIGN, EDLIN, and JOIN) you get the `Incorrect DOS version` error message. You can avoid this problem by loading SETVER before executing the commands. Better still, get the Supplemental DOS 6 disk. The only difference between the DOS 5 versions of the programs and versions on the Supplemental disk is that the new versions do not generate the error message.

Most of these commands will not be missed. I, for one, will not mourn EDLIN. However, you should continue to use MIRROR. Much of the functionality of MIRROR was replaced by UNDELETE, but two important data-protection features are not supported by the standard DOS commands. The MIRROR command creates vital data recovery information and should be used in AUTOEXEC.BAT to make copies of the partition table and copies of the FAT and root directory for each hard drive. For example, the following two commands make a copy of the FAT and root directory for drives C, D, and E and make a copy of the hard disk partition table:

```
mirror c: d: e:
mirror /partn
```

## Summary

DOS version 6.0, arguably the biggest upgrade in DOS history, offers many significant new features. This chapter provided an overview of DOS 6 and highlighted its strengths and weaknesses. The following points were discussed:

- By defining configuration menus and using special boot interruption keys, you can easily control which device drivers and commands are executed at system start-up.
- The new on-line help system provides detailed information on virtually every command and device driver.
- Enhancements to the memory management commands, coupled with the memory optimizer MEMMAKER, give DOS 6 industrial-strength memory management tools.
- By adding DoubleSpace disk compression, defragmentation tools, and a *real* backup program, DOS now provides users with an excellent set of disk management tools.
- The DOS data-protection arsenal was strengthened by the addition of anti-virus tools and enhanced UNDELETE and UNFORMAT programs.
- A number of old commands were dropped, including the DOS 5 MIRROR command. Even though MIRROR is not included in DOS 6, you should continue to use it to protect the FAT and partition table. If you don't have access to DOS 5, you can obtain the old commands by ordering the Supplemental DOS 6 disk from Microsoft.
- The anti-virus, file-undeletion, and backup programs are available in DOS and Windows versions. The versions share similar feature sets, but most Windows users will, no doubt, opt to use the Windows-hosted programs.
- Good as DOS 6 is, it has a number of disappointing weaknesses, such as the lack of support for tape drives, the cumbersome way that compressed diskettes must be handled, and SMARTDRV's inability to cache CD-ROM drives.

Chapter 2 investigates some of the more popular and frequently used commands and explains how to use some lesser-known switches to leverage the most power from these commands.

# Chapter 2

# Mastering the Top DOS Commands

### In This Chapter

- So many commands, so little time
- Getting the best help
- Kicking the COPY habit
- Using the new MOVE command
- Deleting files
- Zapping it with DELTREE
- Using DIR — yes, DIR
- Looking at directory-navigation commands
- Creating a smarter PATH
- Using MSD, the administrator's friend
- Making the DOSSHELL work

## So Many Commands, So Little Time

*"Whoa — what did you just type?"*

Have you ever sat down next to a colleague who's using a PC and seen a command switch being used that you didn't know about? With more than 70 commands and device drivers, and some of them sporting 10 or more switches, it's no wonder that no one remembers them all.

Fortunately, the 80/20 rule applies to DOS — 80 percent of the time, you will use only 20 percent of the commands. This chapter focuses on these most frequently used commands and explains how to get the most productivity from them.

You will also learn about some shareware programs, included on Bob' Shareware Collection disks, that outperform and outsmart their DOS cousins.

## Getting the Best HELP

The DOS 6 on-line help feature is excellent. You may recall that the help table of contents is displayed whenever you enter the HELP command with no switches, as follows:

```
help
```

You can jump directly to a specific topic by executing the HELP command followed by a command name. The following command, for example, displays help for using the CHOICE command:

```
help choice
```

You can also jump directly to a submenu of commands, in addition to the standard commands and device drivers, by entering one of the following help topic strings:

| *Help Topic* | *Description* |
|---|---|
| batch | Batch file commands |
| config | CONFIG.SYS commands |
| international | Commands for international configurations |
| multi-config | Commands for creating a multiboot CONFIG.SYS menu |

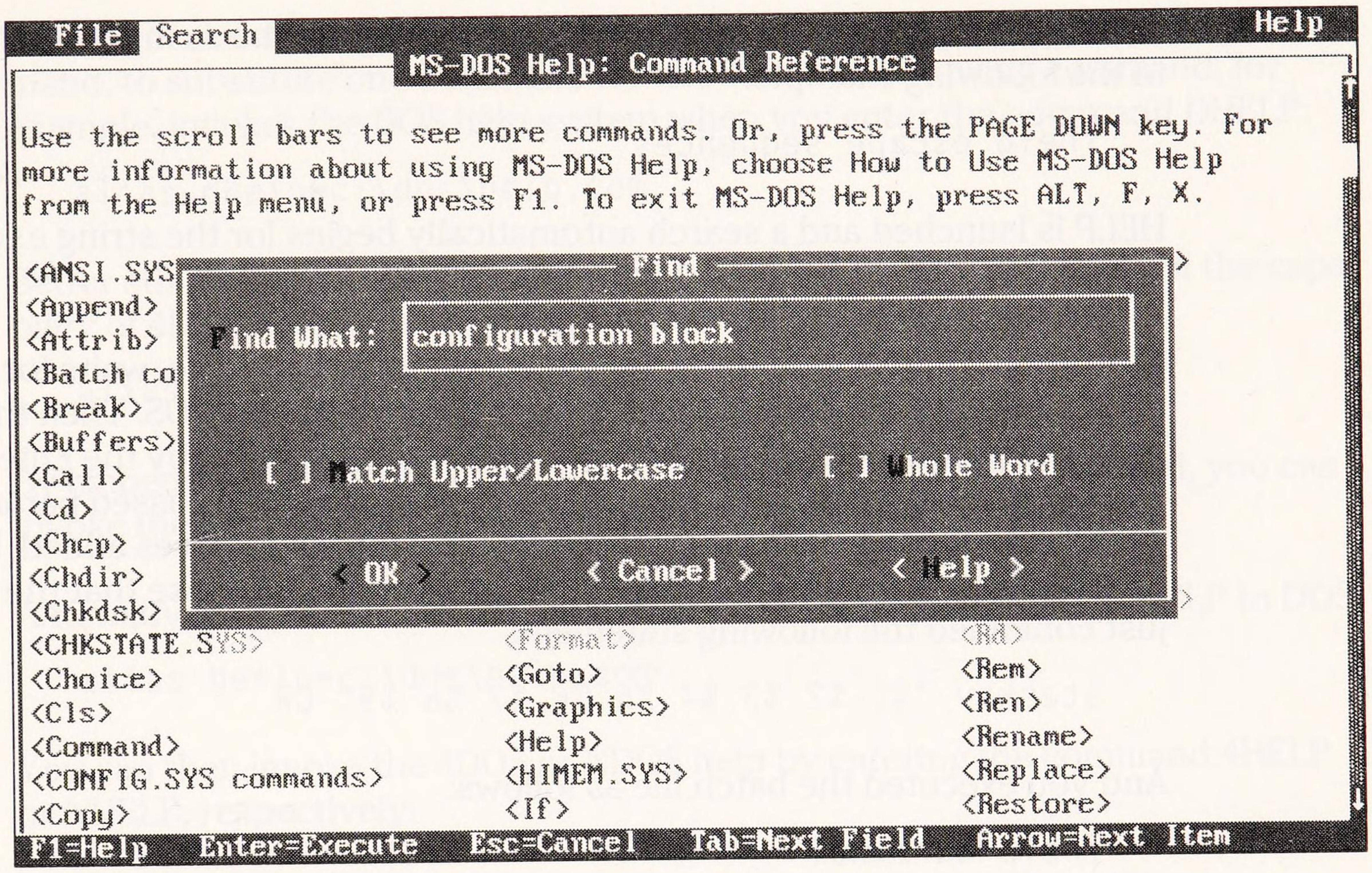
To display a menu that details all the batch file commands, for example, enter the following:

```
help batch
```

### Running help searches

You can search for a specific string in the body of the help text by selecting the **F**ind option from the **S**earch menu. Figure 2-1 shows the Find dialog box that is displayed when you press Alt-S,F. You can enter a string of any length, from a single character to a complete sentence, and you can optionally specify whether the search should be case sensitive. Case-sensitive searches are usually about 20 percent faster than case-insensitive ones.

HELP searches for the string beginning at the cursor position and moving forward through the help file, and accesses each command in alphabetical order. When the last command has been searched, HELP continues searching from the beginning of the help file. HELP displays the search progress in a very subtle manner: In the lower right corner, you will notice the cursor line number being updated as the search progresses. When the text is found, the cursor is positioned beneath the matching text. A full search can take as long as two minutes on some systems (I wish that it would beep when it finds the text).


File
Search
Help
MS-DOS Help: Command Reference
Find
[ ] Match Upper/Lowercase
[ ] Whole Word
< OK >
< Cancel >
< Help >
F1=Help
Enter=Execute
Esc=Cancel
Tab=Next Field
Arrow=Next Item

Use the scroll bars to see more commands. Or, press the PAGE DOWN key. For more information about using MS-DOS Help, choose How to Use MS-DOS Help from the Help menu, or press F1. To exit MS-DOS Help, press ALT, F, X.

Find What: configuration block

```
<ANSI.SYS
<Append>
<Attrib>
<Batch co
<Break>
<Buffers>
<Call>
<Cd>
<Chcp>
<Chdir>
<Chkdsk>
<CHKSTATE.SYS>          <Format>          <Rd>
<Choice>                <Goto>            <Rem>
<Cls>                   <Graphics>        <Ren>
<Command>               <Help>            <Rename>
<CONFIG.SYS commands>   <HIMEM.SYS>       <Replace>
<Copy>                  <If>              <Restore>
```

**Figure 2-1:** HELP's Find dialog box is used to search for a character, word, or phrase.

There is nothing earth-shattering about HELP's search facility. When you combine the search option with STACKEY, however, you can create a quick and very functional help search engine.

The STACKEY utility (included on Bob's Shareware Collection disks), from CtrlAlt Associates, can be used to stuff keystrokes into the keyboard buffer. The following batch file, FHELP.BAT, uses STACKEY to pass a search string to the Find dialog box:

```
@echo off
if "%1"=="" goto plainhelp
stackey W36 @S "F"
if not "%1"=="" if "%2"=="" stackey "%1" CR
if not "%2"=="" if "%3"=="" stackey "%1 %2" CR
if not "%3"=="" if "%4"=="" stackey "%1 %2 %3" CR
if not "%4"=="" if "%5"=="" stackey "%1 %2 %3 %4" CR
if not "%5"=="" if "%6"=="" stackey "%1 %2 %3 %4 %5" CR
if not "%6"=="" if "%7"=="" stackey "%1 %2 %3 %4 %5 %6" CR
if not "%7"=="" if "%8"=="" stackey "%1 %2 %3 %4 %5 %6 %7" CR
if not "%8"=="" if "%9"=="" stackey "%1 %2 %3 %4 %5 %6 %7 %8" CR
if not "%9"=="" stackey "%1 %2 %3 %4 %5 %6 %7 %8 %9" CR
:PLAINHELP
help
```

To use this batch file, just enter FHELP, followed by the string to search for, as in the following example:

```
fhelp escape sequences
```

HELP is launched and a search automatically begins for the string `escape sequences`. Neat.

The first STACKEY command, on line three of the batch file, waits for 2 seconds (W36) and then sends the Alt-S,F keystroke sequence to DOS. Then the passed parameters are stuffed into the keyboard buffer, followed by the Enter key (CR). The series of IF statements are necessary because every passed parameter must be separated by a space; there must be no extra spaces added, however, to separate null (unused) parameters. For example, suppose that the batch file just contained the following statement:

```
stackey "%1 %2 %3 %4 %5 %6 %7 %8 %9" CR
```

And you executed the batch file as follows:

```
fhelp terminal
```

The string passed to the Find dialog box would be `terminal` followed by eight spaces. By using the IF statements as shown in the preceding batch file, the string passed to FHELP would be `terminal` with no spaces after it.

Refer to Part V of this book for more information about STACKEY.

**GEM**

Normally, when you enter the HELP command with a topic name, the name must correspond exactly with the first word of the entry in the main help contents; otherwise, the following message is displayed:

```
Match not found
```

By using the FHELP batch file, you can use a partial name to select a topic. When you enter the following command, for example, the XCOPY command is highlighted:

```
fhelp xco
```

## Solving help conflicts with 4DOS and NDOS

If you use 4DOS (from J. P. Software) or its hybrid, NDOS (from Symantec Corporation), to replace COMMAND.COM as the command interpreter, you may experience problems in accessing the DOS HELP command. When you enter the command HELP, 4DOS (or NDOS) invokes its own internal help rather than DOS's help.

You can maintain access to both help systems by using the 4DOS alias command, to substitute one command for another. The following command, for example, invokes the DOS help system when you enter the command DHELP:

```
alias dhelp=c:\dos\help.com
```

Better still, you can enter the following command to take advantage of the capability of 4DOS to recognize partial commands by using an asterisk:

```
alias dh*elp=c:\dos\help.com
```

By embedding the asterisk after the letters DH in the ALIAS command, you can invoke the DOS help system by entering DH, DHE, DHEL, or DHELP.

Alternatively, you can use the following ALIAS command to assign HELP to DOS:

```
alias he*lp=c:\dos\help.com
```

You can then invoke the 4DOS or NDOS help by entering the command 4HELP or NHELP, respectively.

## Help tricks for Microsoft C programmers

Although you can print every individual help page, there is no convenient way to print the entire help text for all commands. Furthermore, there is no way to customize the help text. No way, that is, unless you have the Microsoft C compiler (version 6 or later).

The entire help text for the HELP command is stored in the file HELP.HLP in the DOS directory. This file was created by using the Microsoft C Helpmake utility. You can use the Helpmake program, included in the BIN directory of the C compiler, to decode the HELP.HLP file and convert it to ASCII text. You can then print the text file or make modifications to the file and re-create a revised binary help file.

To see a summary of Helpmake's switches, enter the following command:

```
helpmake /h
```

The following command creates a file called HELP.TXT that contains the source text for the DOS help:

```
helpmake /d help.hlp /ohelp.txt
```

If you want to make enhancements to the on-line help, you can edit the text file and then use the /E switch to rebuild the HELP file. Refer to the *Microsoft C Environment and Tools* manual for additional information about Helpmake.

## Using the /? switch

The HELP command is not the only way in which DOS provides command assistance. If you enter a command followed by the /? switch, DOS displays a brief summary of the command usage, including a list of all the switches the command supports. The following output, for example, was generated from the command CHOICE /?:

```
Waits for the user to choose one of a set of choices.

CHOICE [/C[:]choices] [/N] [/S] [/T[:]c,nn] [text]

/C[:]choices Specifies allowable keys. Default is YN
/N           Do not display choices and ? at end of prompt string.
/S           Treat choice keys as case sensitive.
/T[:]c,nn    Default choice to c after nn seconds
text         Prompt string to display

ERRORLEVEL is set to offset of key user presses in choices.
```

If you are writing batch files for other users, you should consider writing them to support the /? switch. The following modified version of FHELP.BAT has been enhanced to support /? (the boldface areas indicate the commands that provide the /? support):

```
@echo off
rem Finds a string using DOS HELP command.
if “%1”==”/?” goto commandhelp
if "%1"=="" goto plainhelp
stackey W36 @S "F"
if not "%1"=="" if "%2"=="" stackey "%1" CR
if not "%2"=="" if "%3"=="" stackey "%1 %2" CR
if not "%3"=="" if "%4"=="" stackey "%1 %2 %3" CR
if not "%4"=="" if "%5"=="" stackey "%1 %2 %3 %4" CR
if not "%5"=="" if "%6"=="" stackey "%1 %2 %3 %4 %5" CR
if not "%6"=="" if "%7"=="" stackey "%1 %2 %3 %4 %5 %6" CR
if not "%7"=="" if "%8"=="" stackey "%1 %2 %3 %4 %5 %6 %7" CR
if not "%8"=="" if "%9"=="" stackey "%1 %2 %3 %4 %5 %6 %7 %8" CR
if not "%9"=="" stackey "%1 %2 %3 %4 %5 %6 %7 %8 %9" CR
:PLAINHELP
help
goto quit
:COMMANDHELP
echo Searches for a string in the DOS on-line help.
echo.
echo FHELP [/?] [text]
```

```
echo.
echo When no switches are specified, the help contents are displayed
echo /?    Displays this help text
echo text  Specifies the text to search for
echo.
:QUIT
```

## Using FASTHELP

The FASTHELP command might have been more appropriately named HELPLITE because it provides a brief description of each DOS command.

**NOTE**

FASTHELP used to be called DOSHELP, and you can still enter the command DOSHELP if you prefer. It was renamed to avoid confusion with the new HELP command.

If you enter the FASTHELP command with no switches, nearly five screens full of information that summarize each command are displayed. If you enter the FASTHELP command followed by a command name, more detailed information about the specific command is presented. Behind the scenes, FASTHELP simply executes the command with a /? switch.

**GEM**

The text displayed by FASTHELP is stored in the ASCII file DOSHELP.HLP in the DOS directory. Because HELP now provides the real on-line help for DOS commands, you might consider replacing the data in DOSHELP.HLP with information about your favorite batch files and utilities.

The format of DOSHELP.HLP is straightforward. Any line beginning with the @ character is ignored — it is used to add comments to the file. The help text for commands should be added in alphabetical order, and the command name should begin in the first column. To add more than one line of text, make sure that the second and subsequent lines begin with one or more spaces. Otherwise, DOS treats the line as a new command.

The following custom version of DOSHELP.HLP describes some custom batch files:

```
@ Custom on-line help for batch files. Accessed by FASTHELP
CHKFRAG    Checks the fragmentation of a set of files.
EDITBAT    Launches EDIT and prepares a new batch
           file for editing.
FHELP      Searches for a string in the DOS on-line help.
HEXMATH    Uses DEBUG to provide a simple HEX calculator.
SAVEENV    Saves the active environment. Execute OLDENV to
           subsequently restore the old environment.
SET25      Sets screen into 25 line mode.
SET43      Sets screen into 43 line mode.
SET50      Sets screen into 50 line mode.
```

There is one problem with customizing FASTHELP. Even if you add /? support to your batch files, FASTHELP does not display custom help when you pass the name of the batch file as a FASTHELP switch. For example, when you enter the command

```
fasthelp fhelp
```

DOS responds with the message

```
Help not available for this command
```

FASTHELP calls only COM or EXE files and (stupidly) ignores batch files.

TURBOBAT, which is included on Bob's Shareware Collection disks, is a batch-file compiler that can convert BAT files to COM files. I use TURBOBAT, part of the ExtraDOS Toolbox (from Foley Hi-Tech Systems) all the time. If you compile your batch file and locate the new COM file somewhere on the search path, FASTHELP calls it with the /? switch.

## Applications help

In addition to the on-line help available with the HELP command, the DOS programs in Table 2-1 feature their own extensive help systems.

**Table 2-1 DOS Programs That Have Their Own Help System**

| *Command* | *Description* |
|---|---|
| DBLSPACE | DoubleSpace disk compression |
| DEFRAG | File defragmenter |
| DOSSHELL | Full-screen file and task manager |
| EDIT | Text editor |
| MEMMAKER | Memory optimizer |
| MSAV | DOS anti-virus utility |
| MSBACKUP | DOS backup utility |
| MWAV | Windows anti-virus utility |
| MWBACKUP | Windows backup utility |
| MWUNDEL | Windows undelete tools |
| QBASIC | Basic language interpreter |
| SMARTMON | Windows SMARTDRV monitor |

## Kicking the COPY Habit

Old habits die hard. If you began using DOS many years ago, you are probably a dedicated COPY command user. The truth is that XCOPY is much better than COPY.

From its name, you might think that XCOPY is the former or ex-COPY command. Nothing could be further from the truth. XCOPY, the next generation of COPY, was introduced with DOS 3.2. The name stands for extended *copy*. It's smarter and faster than COPY. Blue Bashers might be interested in knowing that IBM, not Microsoft, programmed XCOPY.

The following list shows some reasons you should use XCOPY rather than COPY:

- XCOPY uses a memory buffer to store multiple files. If you are copying more than one file, XCOPY reads as many files as it can into the buffer; when the buffer fills, XCOPY writes the files to the target location. This technique makes XCOPY much faster than COPY, especially when you are copying from a floppy or remote drive.

  There is one exception to the rule. If you are copying a single file that is less than 64K in size (COPY's buffer size), COPY is faster than XCOPY. This anomaly exists because COPY is an internal command that is already loaded in memory; XCOPY is external and must be loaded in memory before the file is copied.

- You can use the /S switch with XCOPY to copy an entire subdirectory tree.

- If you are copying a large set of files to multiple floppy disks, first use the ATTRIB command to set all the archive bits on (ATTRIB FILEMASK +a, for example) and then use the /M switch with XCOPY. This switch instructs XCOPY to copy only files that have the archive bit set. After each file has been copied, the archive bit is turned off. When the first floppy disk is full, DOS displays this message:

  ```
  Insufficient disk space
  ```

  Just change disks and reissue the same XCOPY command. Because the files that already have been copied have their archive attribute turned off, they are ignored and XCOPY recommences with the correct file.

- Other switches supported by XCOPY but not by COPY are /D, to copy files that have been modified on or after a specified date, and /P, which prompts for confirmation before creating each file copy.

- Use XCOPY in batch files because it returns an ERRORLEVEL indicating whether the copy was a success; COPY does not.

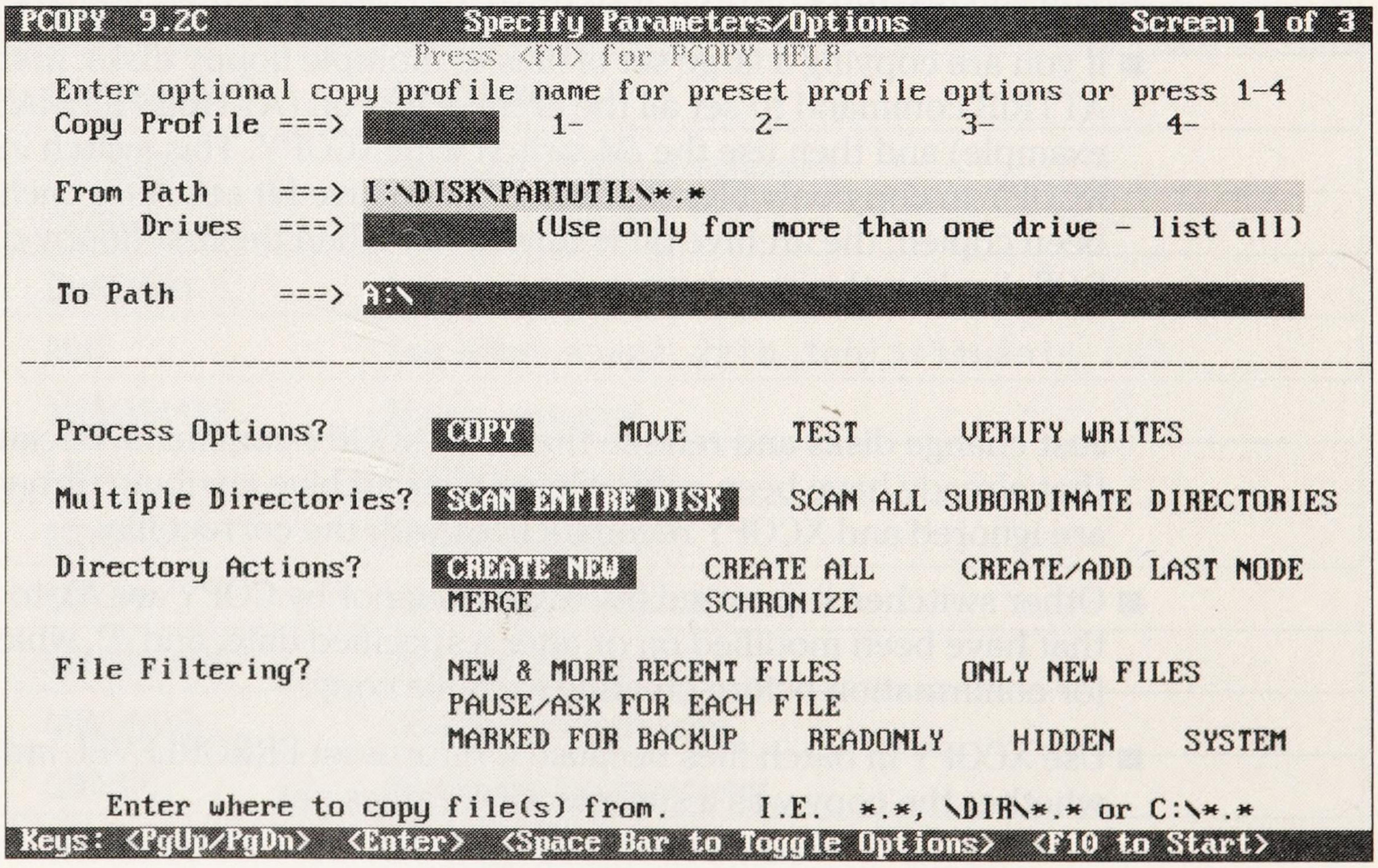
Enough said.

As good as XCOPY is, it pales in comparison to PCOPY. PCOPY is part of the Patriquin Utilities, from Patri-Soft. An evaluation copy is included on Bob's Shareware Collection disks. The following list shows just a few of the PCOPY features:

- Enables you to save older versions of files with new names before replacing them.
- Deletes the original file if /X is specified.
- Enables you to pause processing at any time by pressing any keyboard key. After the program is stopped, you can terminate it by pressing Esc.
- Allows commands to be tested so that you can be sure that commands are specified as you desire.
- Optionally prompts for confirmation before overwriting existing files.

You can execute PCOPY as a command-line utility or in full-screen mode (see Figure 2-2).

The PocketD program D.EXE is also included on Bob's Shareware Collection disks. One of D's primary functions is copying files, and it includes many unique features. Unlike PCOPY, D is solely a command-line utility (although a full-screen utility is under development) and is controlled by using switches. By using a few simple switches, you can invoke some extremely powerful copy operations.



```
PCOPY  9.2C            Specify Parameters/Options              Screen 1 of 3
                          Press <F1> for PCOPY HELP
  Enter optional copy profile name for preset profile options or press 1-4
  Copy Profile ===>            1-         2-         3-         4-

  From Path    ===> I:\DISK\PARTUTIL\*.*
       Drives  ===>          (Use only for more than one drive - list all)

  To Path      ===> A:\

 ---------------------------------------------------------------------------

  Process Options?       COPY      MOVE       TEST        VERIFY WRITES

  Multiple Directories?  SCAN ENTIRE DISK  SCAN ALL SUBORDINATE DIRECTORIES

  Directory Actions?     CREATE NEW       CREATE ALL       CREATE/ADD LAST NODE
                         MERGE            SCHRONIZE

  File Filtering?        NEW & MORE RECENT FILES      ONLY NEW FILES
                         PAUSE/ASK FOR EACH FILE
                         MARKED FOR BACKUP   READONLY   HIDDEN   SYSTEM

     Enter where to copy file(s) from.    I.E.  *.*, \DIR\*.* or C:\*.*
 Keys: <PgUp/PgDn>  <Enter>  <Space Bar to Toggle Options>  <F10 to Start>
```

**Figure 2-2**: You can use PCOPY in full-screen mode to select each switch visually before copying the files.

To copy a series of files to the \STUFF directory, you might issue a command like the following:

```
d *.c *.h *.asm /c \stuff
```

The following command copies all the files in the default directory and in all subdirectories to drive A and creates the same directory structure on the floppy disk:

```
d /cp a:\
```

D has an excellent feature for network users. A common problem in copying a large number of files on a network is that another user has already opened some of the files to be copied. D can skip these files and automatically return to them later and try again. That's smart.

D supports many more copy-related switches; see Part V of this book for more information.

## Using the New MOVE Command

DOS finally has a MOVE command to move one or more files from one directory to another. If the files are moved to another directory on the same drive, MOVE is smart enough to just change each file's directory entries (the file data is not physically moved). When files are moved to a different drive, however, the files are copied to the target location and then the original is deleted.

The syntax for MOVE is similar to COPY but not exactly the same. Refer to Part IV, "The Complete DOS 6 Command Reference," for a full explanation of the syntax.

One nice feature of MOVE is that it allows multiple source filenames to be specified. The following command, for example, moves all files with DOC and SAM to the DOCS directory:

```
move *.doc,*.sam \docs
```

Unlike COPY, MOVE insists that you specify a source filename. You cannot, for example, move all the files in the STUFF directory to the TRASH directory by using the following command:

```
move \stuff \trash
```

MOVE tries to move the STUFF *file* from the root directory to the TRASH directory or (if TRASH doesn't exist) tries to rename the STUFF directory to TRASH. The correct use of the command is as follows:

```
move \stuff\*.* \trash
```

Or, using the briefer dot notation, you can type the following:

```
move \stuff\. \trash
```

Another difference between COPY and MOVE is that MOVE requires you to specify the target directory. You cannot move files to the current directory by omitting the target directory from the command line.

One quirk of the MOVE command, which Microsoft calls a feature, is that it can be used to rename a directory. Well, kinda. You cannot move a directory to another location in the directory hierarchy, but you can assign it a new name. To rename the \TRASH directory to \JUNK, for example, enter the following command:

```
move \trash \junk
```

One final note: If you specify only one file as the source and one file as the target, you can rename the file when it is moved. To move the file IQUIT.DOC from the DOCS directory to RESIGN.DOC in the BADDAY directory, for example, enter the following command:

```
move \docs\iquit.doc \badday\resign.doc
```

We have been waiting ten years for a MOVE command, and I, for one, am disappointed with Microsoft's offering. Both of the excellent shareware programs PocketD and PCOPY provide superior move facilities. These programs provide the same functionality to move operations as they do to copy operations (discussed earlier in this chapter).

By adding the /X switch to any PCOPY command or the M switch to D, the operation instantly becomes a move rather than a copy.

## Deleting Files

There is not much mystery about the DEL (or ERASE) command. Even though DEL supports only one switch, most users don't remember it. The /P switch prompts for confirmation before deleting each file.

Normally, DEL allows you to specify only a single filemask. The following batch file, DELMANY.BAT, however, enables you to delete multiple filemasks with one command:

```
@echo off
for %%f in (%1 %2 %3 %4 %5 %6 %7 %8 %9) do del %%f /p
```

To delete all the files with a BAK or $$$ extension, just enter the following command:

```
delmany *.bak *.$$$
```

Chapter 9 shows how you can use a similar technique to create a DOSKEY macro that deletes multiple files.

When DOS deletes a file, the file's data is not physically deleted; a data recovery program, such as UNDELETE, might be capable of restoring the file. To erase a confidential or classified file and make sure that no one can undelete it, use RELDEL from Tardis DP Consultants. RELDEL, included on Bob's Shareware Collection disks, goes to considerable lengths to ensure that the file's data cannot be recovered.

You can use the PocketD utility D.EXE (discussed earlier in this chapter) to traverse every directory on a hard disk and delete any files that match one or more file specifications. The following command deletes all files with a BAK extension on the current drive:

```
d *.bak /WFzz
```

The following command uses the qz switch to prompt for confirmation before deleting each file; it optionally enables you to view the file before responding:

```
d *.bak *.del *.$$$ *.tmp /WFqz
```

An evaluation copy of PDEL, another program in the Patriquin Utilities (from Patri-Soft), is included on Bob's Shareware Collection disks. PDEL is a command-line utility for deleting files. In addition to all the standard features, it can delete files based on their date or size, begin deleting files when a specific filename is encountered in the directory, stop deleting files when another file is reached, exclude files from specified filemasks, and much more.

The ExtraDOS Toolbox, from Foley Hi-Tech Systems, offers a number of tools that simplify file-deletion chores, including the ones in this list:

| | |
|---|---|
| ALLSUB | Executes any command in the current directory and in all nested subdirectories. (ALLSUB can be used with any command, not just with DEL.) To delete all files beginning with the letters *BUD* on drive C, for example, enter the following command from the root directory of drive C: |

```
allsub del bud*.*
```

| | |
|---|---|
| CLEANUP | A full-screen utility for analyzing a drive, identifying duplicate or unused files, and optionally deleting them. |

NO Enables you to exclude a specified file or groups of files from a command. The following command, for example, deletes all the files in the current directory *except* files with a DOC extension:

```
no *.DOC del *.*
```

Multiple NO statements can be used. The following command, for example, deletes all files except programs and batch files:

```
no *.bat no *.exe no *.com del *.*
```

## Zap It with DELTREE

DOS 6

DOS 6 includes the powerful and destructive DELTREE command, which removes a directory, all its subdirectories, and all the files contained in these directories.

CAUTION

DELTREE can erase thousands of files with one simple command. Always be cautious when you use DELTREE. Stop and think before you press Enter and make sure that you have specified the correct directory path.

You can use the /Y switch to bypass the standard confirmation prompt, but — because of the destructive nature of the command — I don't recommend that you use it.

In DELTREE's normal format, you pass a single pathname that identifies the parent directory to be expunged. The following command, for example, removes all the files in the BACKUP directory and below:

```
deltree c:\backup
```

GEM

Unlike DEL, DELTREE deletes hidden, system, and read-only files. You can delete any single file, regardless of the file's attributes, by specifying a filename rather than a directory path. To delete the hidden file DBLSPACE.BIN in the root directory of drive C, for example, enter the following command:

```
deltree c:\dblspace.bin
```

CAUTION

DELTREE supports filemasks, but you should use them with extreme caution. When a filemask is specified, DELTREE deletes all matching files and all matching directories in addition to *any* files stored in the matching directories.

# DIR — Yes, DIR

Few people take advantage of the switches recently added to the ever-popular DIR command. The full syntax of the DIR command is explained in Part IV of this book, but a few practical tips might illustrate some of the less well-known features.

## Listing subdirectories

I often see experienced DOS users entering the following command to display a list of all the directories in the current directory:

```
dir *.
```

This command displays a list of all (nonsystem and nonhidden) files in the current directory that do not have an extension. Because most users do not create directory names that have extensions, the command lists all the directories. Unfortunately, the command lists any file that doesn't have an extension.

In its day, `dir *.` was a useful shortcut; thanks to the /A (for *a*ttribute) switch, however, there is now a foolproof way to list all directories regardless of their name. Simply use the identifier `d` with the /A switch to indicate that you want to view all files that have the directory attribute set:

```
dir /ad
```

## Displaying all files, including hidden and system files

Another problem with plain DIR is that it normally doesn't list hidden or system files. If you want to see all the files in a directory, just use the /A switch with no identifier, as follows:

```
dir /a
```

You can also specify a filemask with the /A switch. To display all hidden files in the root directory of drive C that have a BIN extension, for example, just enter the following command:

```
dir *.bin /a
```

## Searching for files

My favorite switch of all time is the /S switch for DIR. The /S switch instructs DOS to read the specified directory and all subdirectories when it displays a directory listing.

To search the entire hard drive for a file or files, just specify the appropriate filename or filemask in addition to the /S switch. To search the entire drive C for the file GIRLFRND.BUK, for example, enter the following command:

```
dir \girlfrnd.buk /s
```

Similarly, to list all files that have an extension of ASM, enter the following command:

```
dir \*.asm /s
```

**NOTE**

The new MSD command can search for, and optionally display, any file. This feature is discussed in more detail later, in the section "MSD, the Administrator's Friend."

**DISK**

The use of the /S switch to help find files is useful, but QFIND (from Bruce Gavin), which is included on Bob's Shareware Collection disks, is more flexible and much, much faster. QFIND searches for any file or filemask on the current drive or all drives, and does so at breakneck speed. If a lot of files are found, you can scroll through a list of all matching files. You can even search for files that have a certain set of characters somewhere in the filename. The following command, for example, displays all the files that have *bud* in their name:

```
qfind *bud*.*
```

## Creating a smarter filemask

**GEM**

If a filemask begins with an asterisk, DOS assumes that you want to list *all* files. If you enter the following command, for example, DOS lists all files, not just the ones with the characters 93 in their name:

```
dir \*93*.* /s
```

By using the FIND filter, you can work around this DOS shortcoming. FIND searches for any string anywhere on a line. By reading the output from DIR, FIND displays only the names of files that contain the specified string.

The FIND command is explained fully in Chapter 10.

The following batch file, FLEXDIR.BAT (for *flex*ible *dir*ectory), uses this technique and provides a convenient way to search for files when you can remember only part of the name:

```
@echo off
if "%1"=="" goto noparam
echo Searching for matching files....
dir *.* %2 %3 %4 %5 %6 %7 %8 %9|find "%1" /i
goto quit
:NOPARAM
dir *.* /a
:QUIT
```

To display all files in the current directory that include the characters 93, you enter the following command:

```
flexdir 93
```

To search the entire drive, you enter the following command from the root directory:

```
flexdir 93 /s
```

## Searching for a partial directory name

By using a technique similar to the flexible directory lister discussed in the preceding section, you can search for any directory that contains a specified set of characters.

The batch file FINDDIR.BAT uses the /S and /AD switches with DIR to list all directories on the drive. The output from DIR is redirected to FIND, which searches for any line that contains the word *Directory*. A second FIND command searches these lines for the user-specified characters. Here are the contents of FINDDIR.BAT:

```
@echo off
echo Searching for directories....
if "%1"=="" goto noparam
dir \ /s /ad | find "Directory" | find "%1" /i
goto quit
:NOPARAM
dir \ /s /ad | find "Directory"
:QUIT
```

The following command shows FINDDIR being used to search for any directory that contains the characters ICO:

```
finddir ico
```

FINDDIR responds, on my system, by displaying the following output:

```
Searching for directories....
Directory of C:\123W\GRAPHICO
Directory of C:\123W\SHEETICO
Directory of C:\AMIPRO\ICONS
Directory of C:\WINDOWS\ICONS
```

## Setting default DIR switches

By assigning specific DIR switches to the environment variable DIRCMD, you can define your own custom defaults. To always sort your directory listings in date order, for example, with the newest files first, you ordinarily use a command like this one:

```
dir /o-d
```

You can make the /O-D switch the default by adding the following command to AUTOEXEC.BAT:

```
set dircmd=/o-d
```

Whenever you enter DIR, the /O-D switch then is applied.

On the odd occasion when you don't want the default to apply, you should precede the entire switch with the minus character (–). To instruct DIR to ignore the DIRCMD setting and display an unsorted directory listing, for example, you enter the following command:

```
dir /-o
```

I find this override approach to be unnecessarily awkward, especially when multiple switches have been set. The following PDIR.BAT (for *p*lain *DIR*) batch file temporarily disables the DIRCMD settings, executes a standard DIR command that ignores the defaults, and then reactivates the default settings:

```
@echo off
if "%dircmd%"=="" goto notset
set dirtemp=%dircmd%
set dircmd=
dir %1 %2 %3 %4 %5 %6 %7 %8 %9
set dircmd=%dirtemp%
set dirtemp=
goto quit
```

```
:NOTSET
dir %1 %2 %3 %4 %5 %6 %7 %8 %9
:QUIT
```

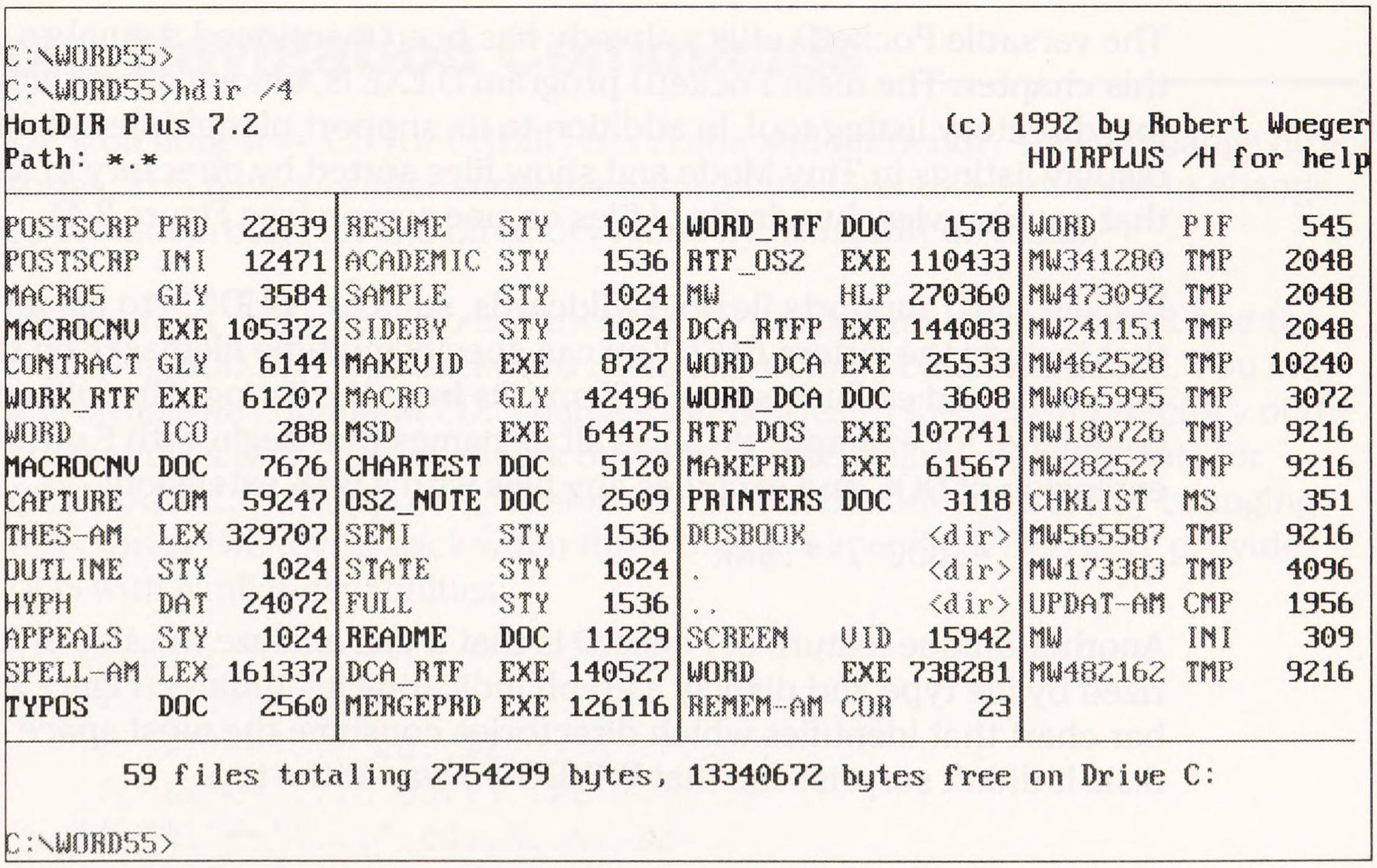
The following example command uses PDIR to display a directory listing in wide format and pause after every screen:

```
pdir /p /w
```

## Shareware alternatives to DIR

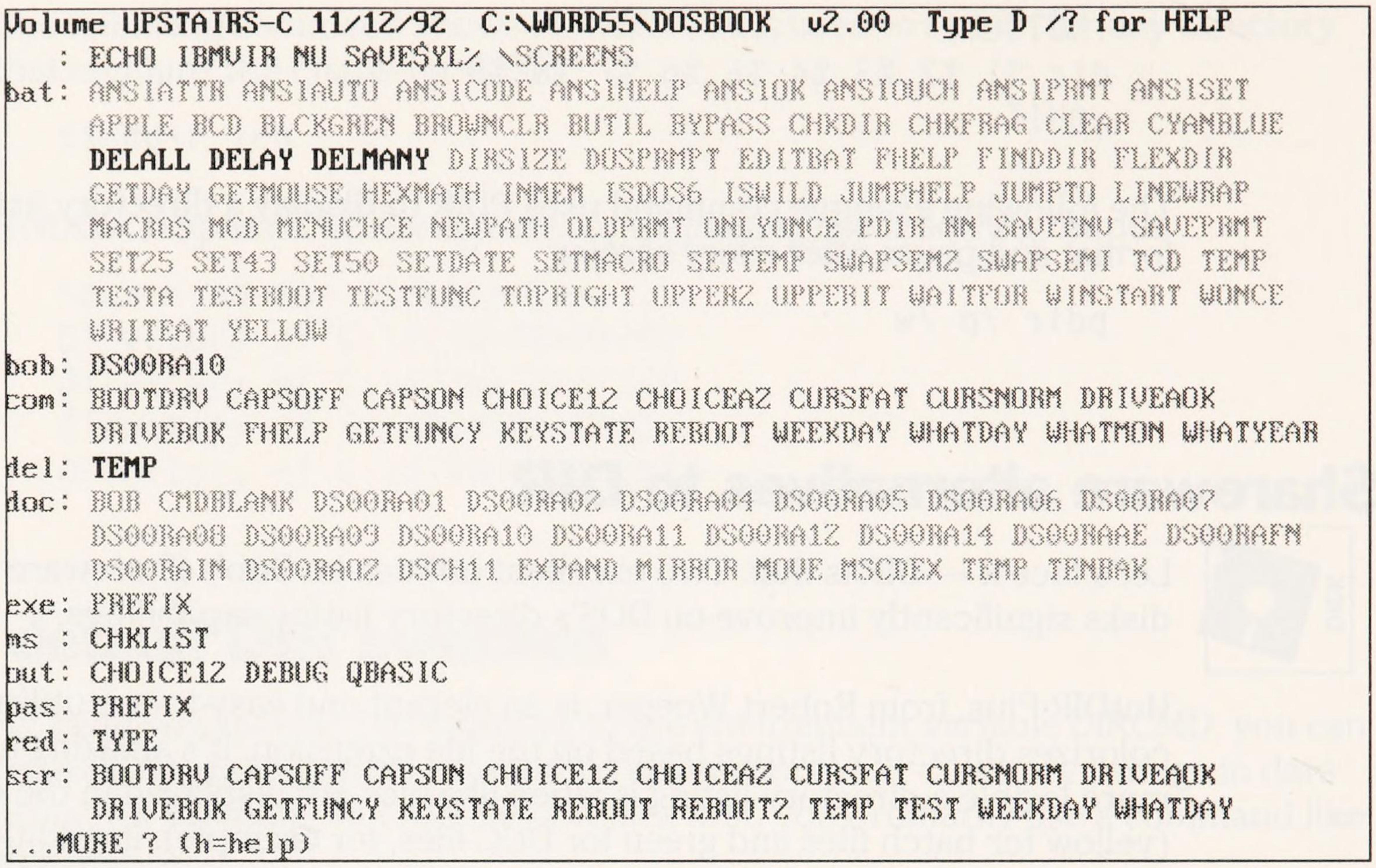
Let's face it — DIR is dull. Two excellent utilities on Bob's Shareware Collection disks significantly improve on DOS's directory listing capabilities.

HotDIR Plus, from Robert Woeger, is an elegant and easy-to-use utility that colorizes directory listings based on the file extension. It's amazing how much more legible a directory listing is when like-files are displayed in the same color (yellow for batch files and green for DOC files, for example). In addition to colorizing the output, HotDIR Plus can sort files, and it supports column formats that range from one to six columns (see Figure 2-3).



```
C:\WORD55>
C:\WORD55>hdir /4
HotDIR Plus 7.2                                          (c) 1992 by Robert Woeger
Path: *.*                                                    HDIRPLUS /H for help

POSTSCRP PRD  22839│RESUME   STY    1024│WORD_RTF DOC   1578│WORD     PIF    545
POSTSCRP INI  12471│ACADEMIC STY    1536│RTF_OS2  EXE 110433│MW341280 TMP   2048
MACROS   GLY   3584│SAMPLE   STY    1024│MW       HLP 270360│MW473092 TMP   2048
MACROCNV EXE 105372│SIDEBY   STY    1024│DCA_RTFP EXE 144083│MW241151 TMP   2048
CONTRACT GLY   6144│MAKEVID  EXE    8727│WORD_DCA EXE  25533│MW462528 TMP  10240
WORK_RTF EXE  61207│MACRO    GLY   42496│WORD_DCA DOC   3608│MW065995 TMP   3072
WORD     ICO    288│MSD      EXE   64475│RTF_DOS  EXE 107741│MW180726 TMP   9216
MACROCNV DOC   7676│CHARTEST DOC    5120│MAKEPRD  EXE  61567│MW282527 TMP   9216
CAPTURE  COM  59247│OS2_NOTE DOC    2509│PRINTERS DOC   3118│CHKLIST  MS     351
THES-AM  LEX 329707│SEMI     STY    1536│DOSBOOK       <dir>│MW565587 TMP   9216
OUTLINE  STY   1024│STATE    STY    1024│.             <dir>│MW173383 TMP   4096
HYPH     DAT  24072│FULL     STY    1536│..            <dir>│UPDAT-AM CMP   1956
APPEALS  STY   1024│README   DOC   11229│SCREEN   VID  15942│MW       INI    309
SPELL-AM LEX 161337│DCA_RTF  EXE  140527│WORD     EXE 738281│MW482162 TMP   9216
TYPOS    DOC   2560│MERGEPRD EXE  126116│REMEM-AM COR     23│

       59 files totaling 2754299 bytes, 13340672 bytes free on Drive C:

C:\WORD55>
```

**Figure 2-3:** HotDIR Plus displays a colored directory in four-column format.



```
Volume UPSTAIRS-C 11/12/92  C:\WORD55\DOSBOOK  v2.00  Type D /? for HELP
   : ECHO IBMVIR NU SAVE$YL% \SCREENS
bat: ANSIATTR ANSIAUTO ANSICODE ANSIHELP ANSIOK ANSIOUCH ANSIPRMT ANSISET
     APPLE BCD BLCKGREN BROWNCLR BUTIL BYPASS CHKDIR CHKFRAG CLEAR CYANBLUE
     DELALL DELAY DELMANY DIRSIZE DOSPRMPT EDITBAT FHELP FINDDIR FLEXDIR
     GETDAY GETMOUSE HEXMATH INMEM ISDOS6 ISWILD JUMPHELP JUMPTO LINEWRAP
     MACROS MCD MENUCHCE NEWPATH OLDPRMT ONLYONCE PDIR RM SAVEENV SAVEPRMT
     SET25 SET43 SET50 SETDATE SETMACRO SETTEMP SWAPSEM2 SWAPSEMI TCD TEMP
     TESTA TESTBOOT TESTFUNC TOPRIGHT UPPER2 UPPERIT WAITFOR WINSTART WONCE
     WRITEAT YELLOW
bob: DS00RA10
com: BOOTDRV CAPSOFF CAPSON CHOICE12 CHOICEAZ CURSFAT CURSNORM DRIVEAOK
     DRIVEBOK FHELP GETFUNCY KEYSTATE REBOOT WEEKDAY WHATDAY WHATMON WHATYEAR
del: TEMP
doc: BOB CMDBLANK DS00RA01 DS00RA02 DS00RA04 DS00RA05 DS00RA06 DS00RA07
     DS00RA08 DS00RA09 DS00RA10 DS00RA11 DS00RA12 DS00RA14 DS00RAAE DS00RAFN
     DS00RAIN DS00RA0Z DSCH11 EXPAND MIRROR MOVE MSCDEX TEMP TEMBAK
exe: PREFIX
ms : CHKLIST
out: CHOICE12 DEBUG QBASIC
pas: PREFIX
red: TYPE
scr: BOOTDRV CAPSOFF CAPSON CHOICE12 CHOICEAZ CURSFAT CURSNORM DRIVEAOK
     DRIVEBOK GETFUNCY KEYSTATE REBOOT REBOOT2 TEMP TEST WEEKDAY WHATDAY
...MORE ? (h=help)
```

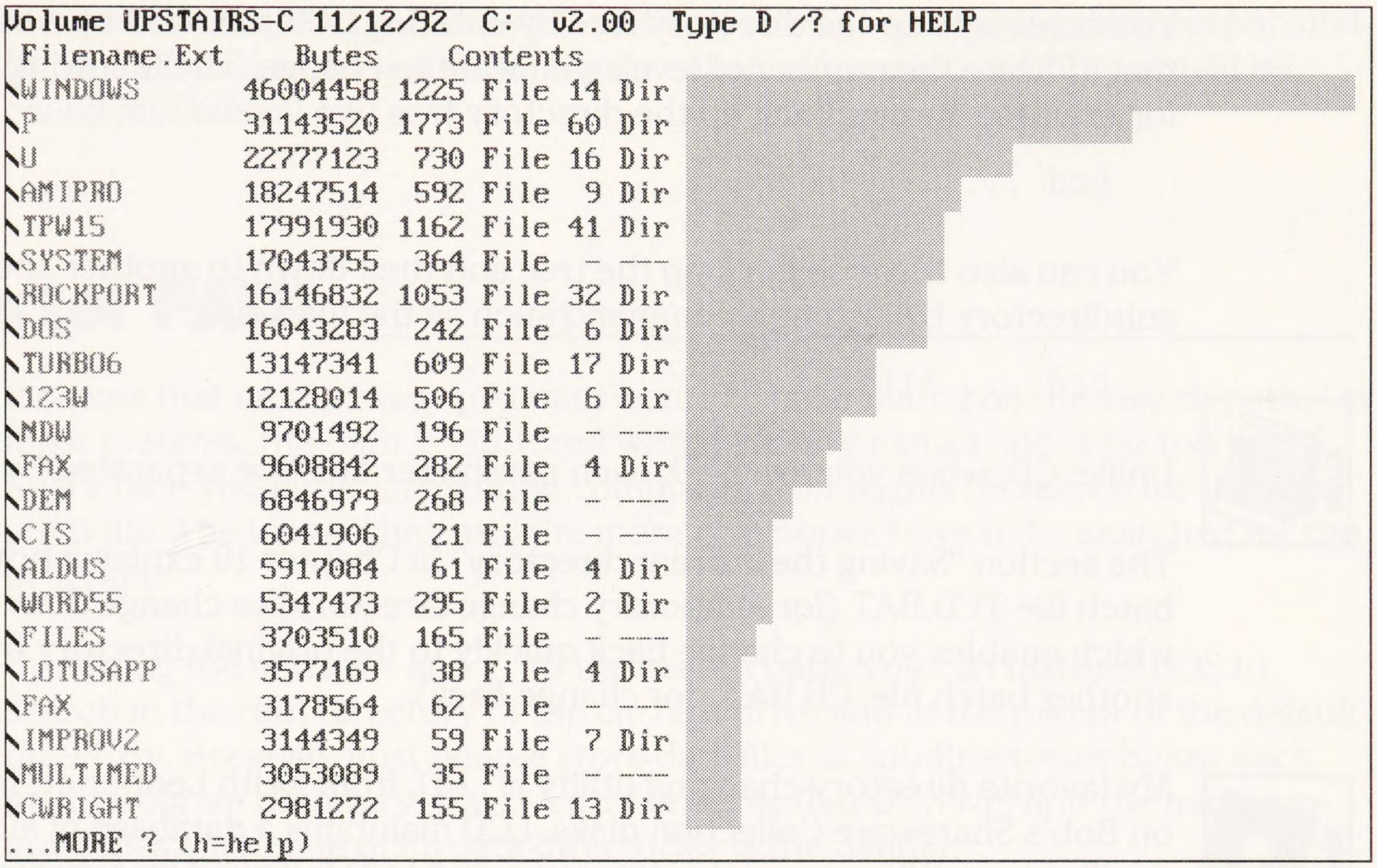
**Figure 2-4:** PocketD can display a large number of files in a compact manner by using Tiny Mode.

The versatile PocketD utility already has been mentioned a number of times in this chapter. The main PocketD program D.EXE is, among other things, a flexible directory listing tool. In addition to its support of multiple columns, D can display listings in Tiny Mode and show files sorted by directory in such a way that you can view hundreds of files on one screen (see Figure 2-4).

PocketD also supports flexible wildcards, such as *BUD*.*, to list all filenames that include the letters *BUD*. You can specify multiple filemasks on the command line and exclude specific filemasks from the listing. The following command, for example, displays all filenames that begin with F or that have an extension of DOC and excludes any files with a BAK extension:

```
d F* *.DOC /-*.BAK
```

Another unique feature of PocketD is that it can analyze the size of files categorized by file type and display a graph indicating its findings. Figure 2-5 shows a bar chart that identifies which directories consume the most space on a hard disk. It didn't surprise me that WINDOWS was at the top.



```
Volume UPSTAIRS-C 11/12/92  C:  v2.00  Type D /? for HELP
 Filename.Ext      Bytes    Contents
\WINDOWS        46004458 1225 File 14 Dir
\P              31143520 1773 File 60 Dir
\U              22777123  730 File 16 Dir
\AMIPRO         18247514  592 File  9 Dir
\TPW15          17991930 1162 File 41 Dir
\SYSTEM         17043755  364 File  - ---
\ROCKPORT       16146832 1053 File 32 Dir
\DOS            16043283  242 File  6 Dir
\TURBO6         13147341  609 File 17 Dir
\123W           12128014  506 File  6 Dir
\MDW             9701492  196 File  - ---
\FAX             9608842  282 File  4 Dir
\DEM             6846979  268 File  - ---
\CIS             6041906   21 File  - ---
\ALDUS           5917084   61 File  4 Dir
\WORD55          5347473  295 File  2 Dir
\FILES           3703510  165 File  - ---
\LOTUSAPP        3577169   38 File  4 Dir
\FAX             3178564   62 File  - ---
\IMPROV2         3144349   59 File  7 Dir
\MULTIMED        3053089   35 File  - ---
\CWRIGHT         2981272  155 File 13 Dir
...MORE ? (h=help)
```

**Figure 2-5:** A bar chart generated by PocketD shows the amount of space used by files in every directory.

## Directory Navigation Commands

Let's be honest — CD (or CHDIR) is a crude and unfriendly way to change directories. You have to either know the name of the directory you want to change to or hunt around for the directory name by using DIR or TREE.

There aren't many ways to enhance the CD command, but you might find the following BCD.BAT (for *b*ackward *c*hange *d*irectory) batch file useful. You may know that the command CD.. changes directories to the parent directory of the current directory. Some network operating systems and replacements for COMMAND.COM support extensions to the double-dot shortcut by changing directories two levels back when three dots are specified. BCD.BAT provides DOS with similar capabilities:

```
@echo off
if "%1"=="" cd
if "%1"==".." cd..%2
if "%1"=="..." cd..\..%2
if "%1"=="...." cd..\..\..%2
if "%1"=="....." cd..\..\..\..%2
```

You can move up the directory tree by entering BCD followed by a series of dots that indicate the number of levels you want to change. The following command, for example, moves back up the directory tree two levels:

```
bcd ...
```

You can also traverse back up the tree and then down to another specific subdirectory by issuing a command such as the following:

```
bcd ... \this\that
```

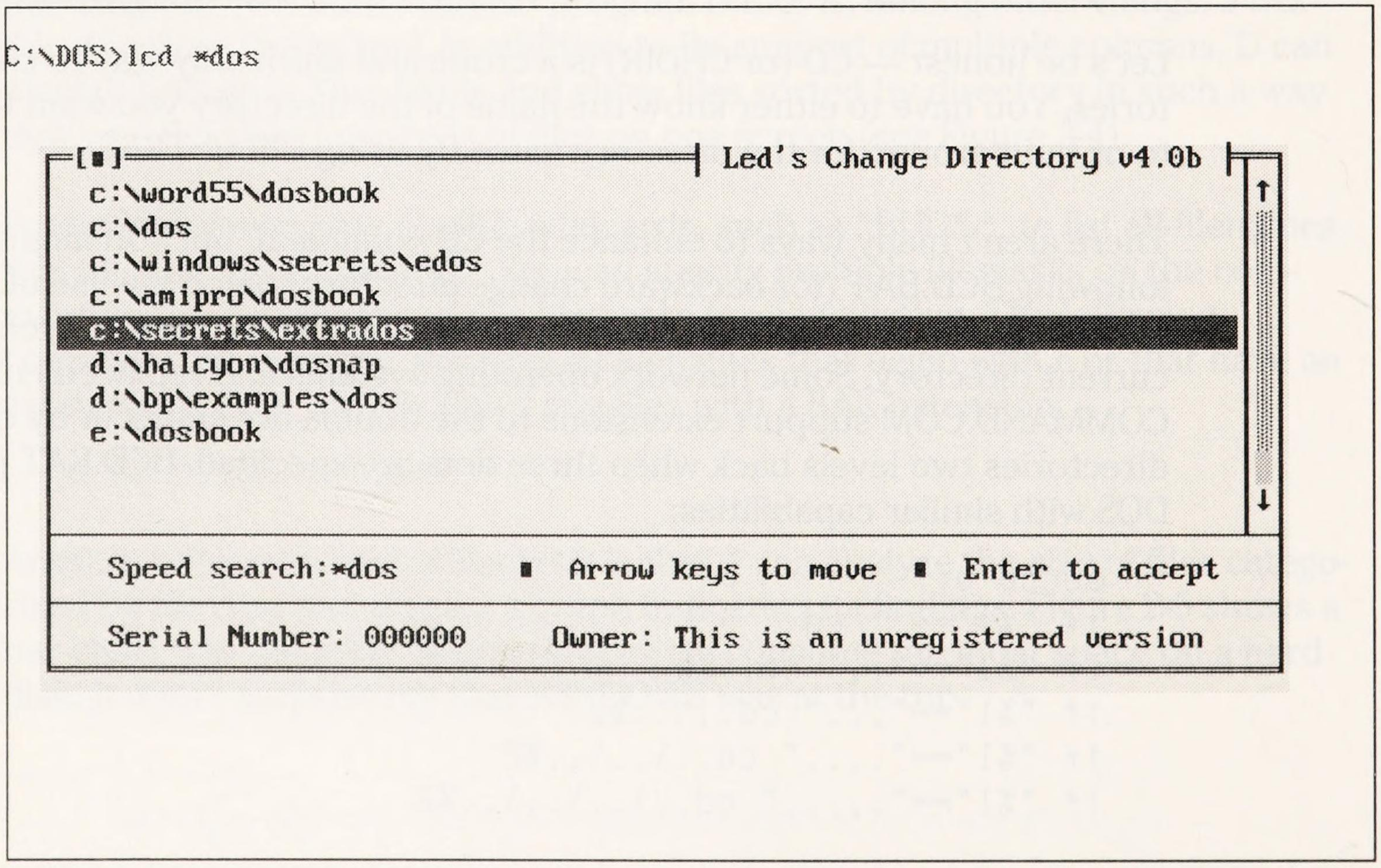
NOTE

Unlike CD, when you use BCD, each parameter must be separated by a space.

The section "Saving the current directory" in Chapter 10 explains how to use the batch file TCD.BAT (for *t*emporary *c*hange *d*irectory) to change directories, which enables you to change back quickly to the original directory by executing another batch file, CB.BAT (for *c*hange *b*ack).

DISK

My favorite directory-changing utility is LCD, from Keith Ledbetter; it is included on Bob's Shareware Collection disks. LCD maintains a database of all directories on all drives. When you enter the command LCD followed by a directory name, or even by a partial directory name, LCD changes to the appropriate directory. If you enter a partial directory name and more than one match is found, a pop-up dialog box lists all the matching directory names (see Figure 2-6). Just choose the directory you want from the list, and it is made the default directory.



```
C:\DOS>lcd *dos

[■]                                Led's Change Directory v4.0b
  c:\word55\dosbook
  c:\dos
  c:\windows\secrets\edos
  c:\amipro\dosbook
  c:\secrets\extrados
  d:\halcyon\dosnap
  d:\bp\examples\dos
  e:\dosbook

  Speed search:*dos      ■ Arrow keys to move  ■ Enter to accept

  Serial Number: 000000     Owner: This is an unregistered version
```

**Figure 2-6**: LCD shows all directories with the characters *DOS* somewhere in the directory name.

NOTE

McAfee Associates (best known for its anti-virus products) is taking responsibility for LCD and actively developing an enhanced version of LCD that will be called MCD.

## A Smarter PATH

It seems that nearly every program wants to be included on the search path. On most systems, the path is cluttered with directory names and is far too long. Every time you enter an external command, DOS begins its search for the program file. The longer the path, the more directories have to be searched for the program.

By adding the entries \ and ..\ to the search path, you can instruct DOS to search in the root directory of the current drive and in the parent of the default directory. Because most people store data files in subdirectories below each main program directory, the ..\ entry ensures that DOS will find the main program files when one of its data directories is the default.

The following entry from AUTOEXEC.BAT uses this technique:

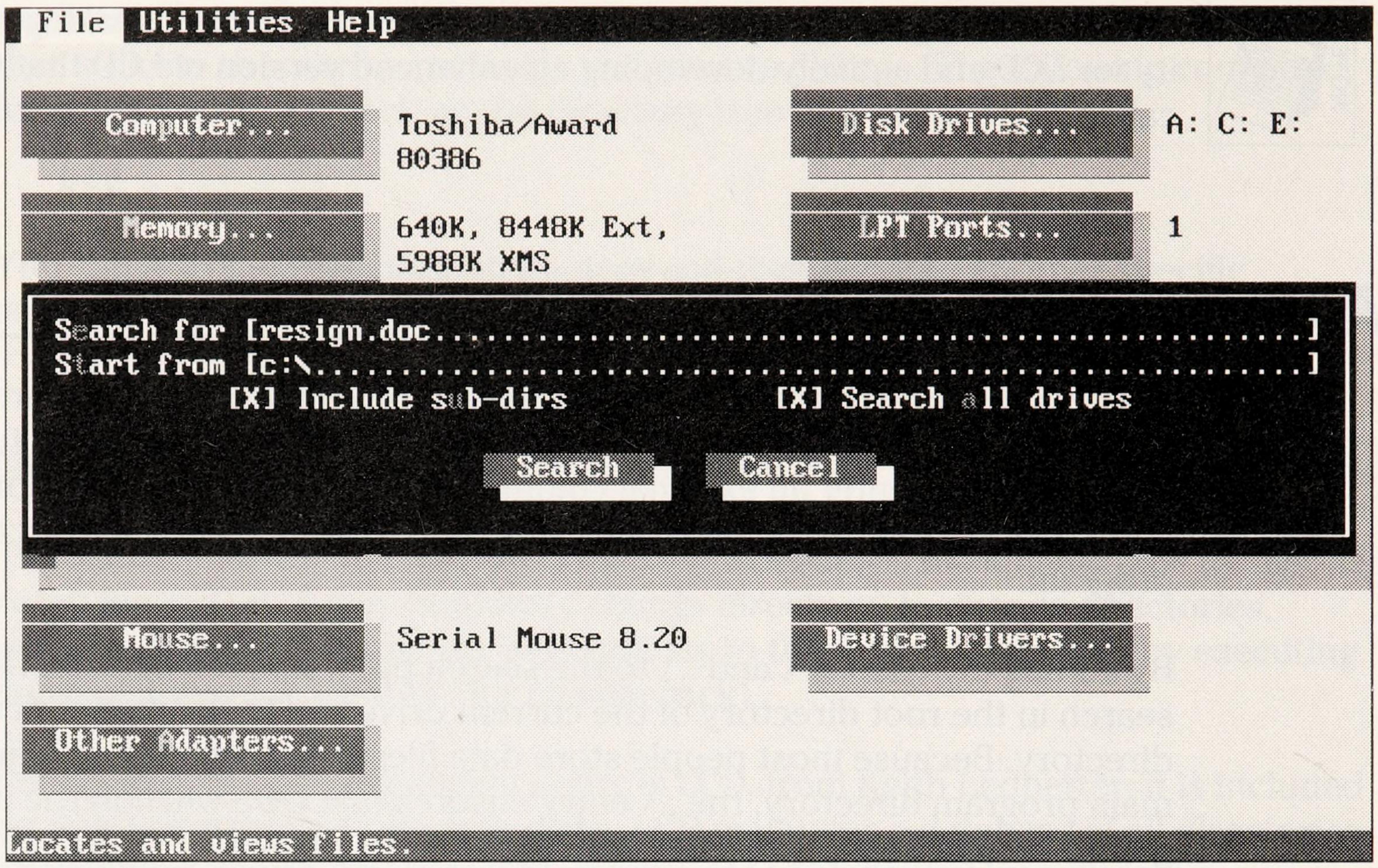
```
path c:\dos;c:\windows;c:\windows\ndw;c:\batfiles;\;..\
```

## MSD, the Administrator's Friend

The DOS program MSD, short for Microsoft System Diagnostics, provides valuable information about your system's hardware and software configuration. MSD displays information about the system board, memory, video, operating system, mouse, disk drives, ports, and related items.

Although MSD is most often used interactively by a user trying to solve a problem, you can also use it from the command line to generate detailed configuration reports. This feature makes it ideal for technical-support staff members who have to diagnose someone else's system problem over the phone or by electronic mail.

Used with the /F switch, MSD analyzes the system and writes a detailed report on the complete system configuration, in addition to the contents of CONFIG.SYS, AUTOEXEC.BAT, and the main INI files, such as WIN.INI, SYSTEM.INI, and DBLSPACE.INI. Before the report is created, MSD prompts the user for a name, an address, and a one-line comment. These details are written at the top of the report.



```
Toshiba/Award
80386
640K, 8448K Ext,
5988K XMS
A: C: E:
1
Search for [resign.doc............................................]
Start from [c:\...................................................]
[X] Include sub-dirs          [X] Search all drives
Serial Mouse 8.20
Locates and views files.
```

Figure 2-7: MSD can search for files across multiple drives.

The following command, for example, prompts for the user details and then generates a detailed report in the ASCII file MSD.OUT:

```
msd /fmsd.out
```

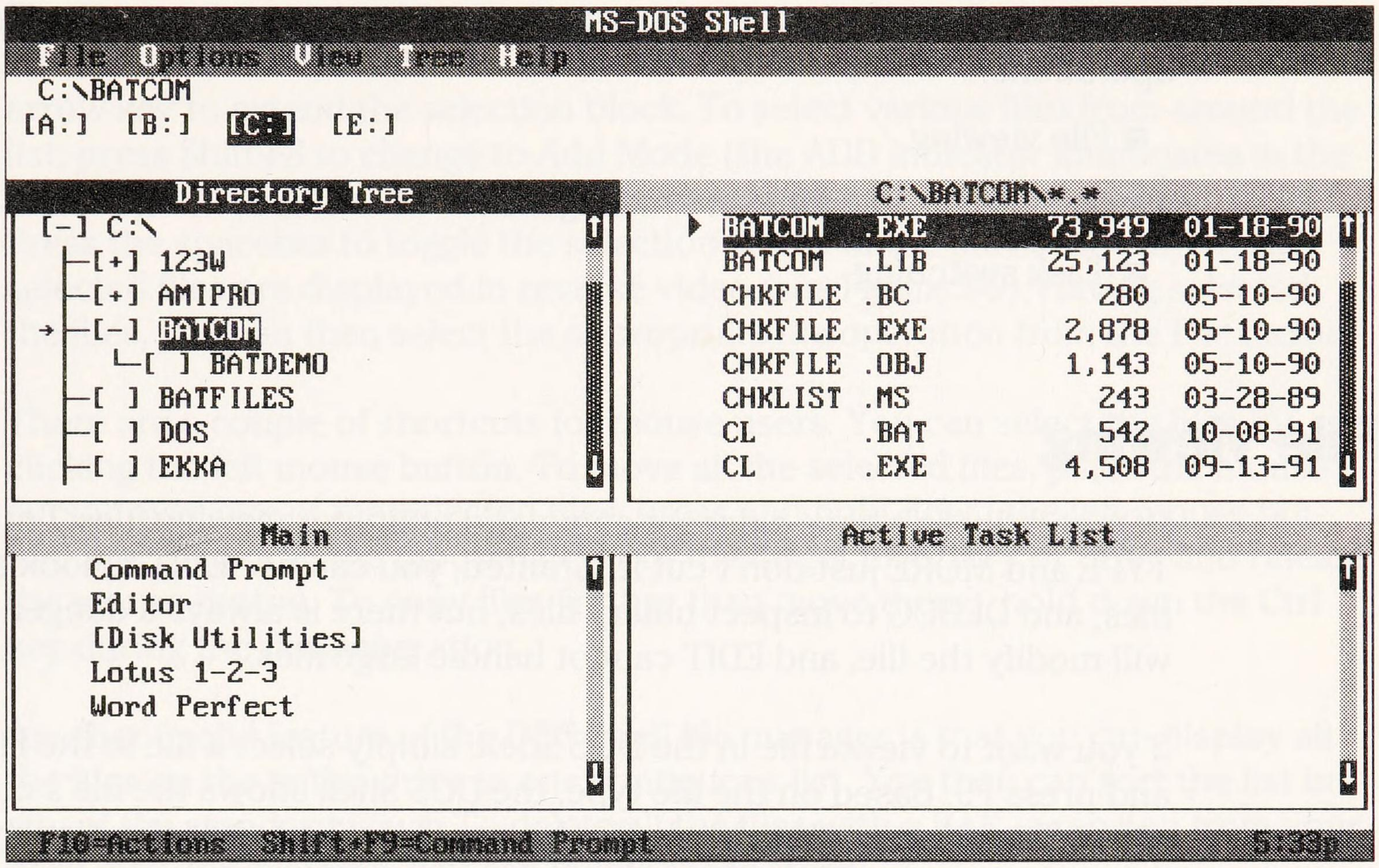
Having entered this MSD command, the user can then send the MSD.OUT file to you by mail or e-mail for analysis.

**GEM**

One surprising and little-known feature of MSD is that it includes a decent file-finding and -browsing tool. You can search for any file by first selecting File⇨Find File from the main MSD menu and then entering the file details in the dialog box (see Figure 2-7). All the files MSD finds during the search are displayed in a scrollable list. You can view the contents of any file by selecting it from the list.

## Making the DOSSHELL Work

The DOS Shell was introduced with DOS 5.0, and Microsoft tried hard to convince users that it was a major new feature. The DOS Shell provides file-management, program-launching, and task-switching capabilities from a full-screen application that supports the mouse. Figure 2-8 shows a typical DOS Shell display with a menu bar and four activity windows.



```
                              MS-DOS Shell
 File  Options  View  Tree  Help
 C:\BATCOM
[A:]  [B:]  [C:]  [E:]

          Directory Tree                          C:\BATCOM\*.*
 [-] C:\                               BATCOM   .EXE     73,949  01-18-90
   ├─[+] 123W                          BATCOM   .LIB     25,123  01-18-90
   ├─[+] AMIPRO                        CHKFILE  .BC         280  05-10-90
 → ├─[-] BATCOM                        CHKFILE  .EXE      2,878  05-10-90
   │  └─[ ] BATDEMO                    CHKFILE  .OBJ      1,143  05-10-90
   ├─[ ] BATFILES                      CHKLIST  .MS         243  03-28-89
   ├─[ ] DOS                           CL       .BAT        542  10-08-91
   ├─[ ] EKKA                          CL       .EXE      4,508  09-13-91

               Main                             Active Task List
     Command Prompt
     Editor
     [Disk Utilities]
     Lotus 1-2-3
     Word Perfect

 F10=Actions  Shift+F9=Command Prompt                                5:33p
```

**Figure 2-8:** The primary interface of the DOS Shell program.

The DOS Shell was Microsoft's attempt to shield users from the terse command prompt and provide an easy-to-use environment for running programs and managing files. In reality, however, only a small percentage of users embraced the new program.

There are two main reasons for the user community's lackluster response: The program is slow, and it arrived too late, at a time when Windows was gaining popularity. Windows certainly provides all the features of DOSSHELL and then some. It is interesting to note that the DOS Shell is no longer being pushed by Microsoft, and it has not been significantly enhanced for DOS 6. Furthermore, the DOS Shell gets less than three pages in the *MS-DOS User's Guide*. Microsoft is clearly placing less emphasis on the DOS Shell. I hope that it's because it is working on providing improved file management and task switching in the core of a future version of DOS. Time will tell.

Some people used to use the DOS Shell solely to perform tasks that couldn't be performed with other DOS commands — namely, renaming directories and moving files. Now these tasks can be performed by the use of the new MOVE command.

If you use Windows, it is highly unlikely that you will need the DOS Shell. There are still some things, however, the DOS Shell does better than plain DOS does. If you don't use Windows, you might be tempted to use the DOS Shell for these operations:

- File viewing
- Diverse file selection and management
- Task switching

## File viewing

DOS still doesn't have a decent command utility for viewing files. I'm afraid that TYPE and MORE just don't cut it. Granted, you can use EDIT to look at ASCII files, and DEBUG to inspect binary files, but there is always a danger that you will modify the file, and EDIT cannot handle large files.

If you want to view a file in the DOS Shell, simply select a file in the File Window and press F9. Based on the file type, the DOS Shell shows the file's contents in either Text Mode or Hex Mode (for binary files). To toggle the display to the other mode, press F9 again.

Bob's Shareware Collection disks include the excellent file-viewing program LIST, from Buerg Software. LIST can quickly display any file of any size, and it offers features such as an optional ruler bar, customizable colors, user-placed bookmarks, text searching, printing, and condensed mode display. To put it bluntly, LIST is the king of file-viewing utilities.

## Diverse file selection and management

The standard DOS commands enable you to manage a block of files only if they all conform to a common filemask. There is no way, for example, to copy with one DOS command all the files that have an EXE, a COM, and a BAT extension. The DOS Shell enables you to select a mixed set of files regardless of the filemask and then perform common file operations such as copying, moving, and deleting them.

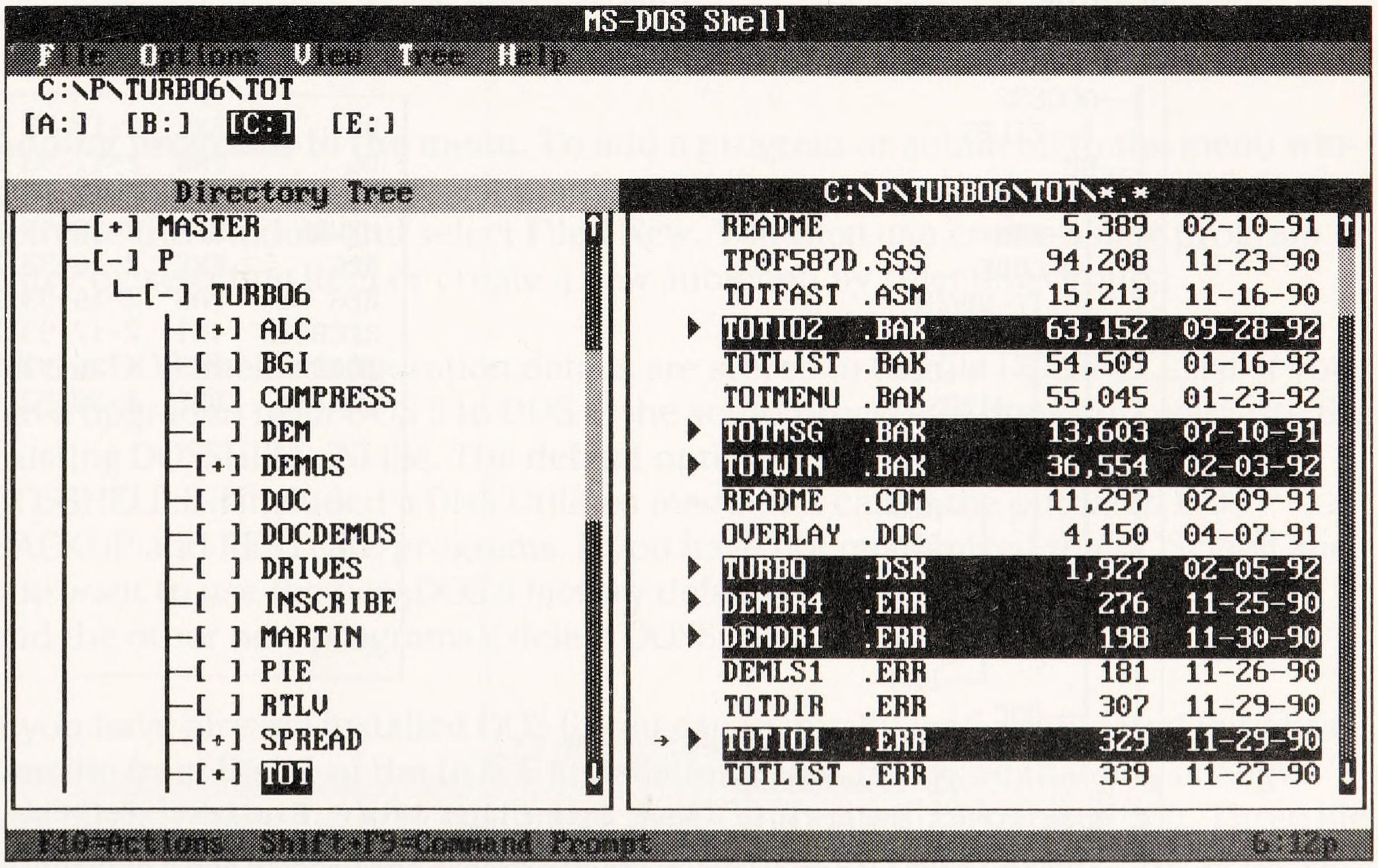
If you are planning to use the DOS Shell for file management, make the file and tree windows larger by choosing **V**iew⇨**S**ingle File List from the DOS Shell menu. Select the menu option **O**ptions⇨**F**ile Display Options to work with files that match a particular filemask. Otherwise, leave it set at *.*. Also, if you want to be able to select a set of files that span more than one directory, you must select **O**ptions⇨Select **A**cross Directories and make sure that a dot appears next to the menu item to indicate that the option is active.

To select a contiguous block of files from the list, move the cursor to the first or last file in the block, hold down the Shift key, and press either the up- or down-arrow key to extend the selection block. To select various files from around the list, press Shift-F8 to change to Add Mode (the ADD indicator illuminates in the lower right corner) and then press the arrow keys to navigate around the list. Press the spacebar to toggle the selection status of the file at the cursor. All selected files are displayed in reverse video (see Figure 2-9). Having selected the files, you can then select the appropriate file operation from the **F**ile menu.

There are a couple of shortcuts for mouse users. You can select the files by clicking the left mouse button. To move all the selected files, place the mouse cursor over one of the selected files, press and hold down the left mouse button, and then drag the mouse cursor to the target directory or drive and release the mouse button. To copy files (rather than move them), hold down the Ctrl key during the drag operation.

Another useful feature of the DOS Shell file manager is that you can display all the files on the entire drive in one contiguous list. You then can sort the list in any of the standard ways. To delete all the files with a BAK extension from your hard disk, for example, you can follow these steps:

1. Choose **O**ptions⇨**F**ile Display Options and then enter `*.BAK`.
2. Choose **V**iew⇨**A**ll Files.



```
                               MS-DOS Shell
 File  Options  View  Tree  Help
 C:\P\TURBO6\TOT
[A:]  [B:]  [C:]  [E:]

        Directory Tree                         C:\P\TURBO6\TOT\*.*
   ├─[+] MASTER                     ↑         README             5,389  02-10-91 ↑
   ├─[-] P                                    TP0F587D.$$$      94,208  11-23-90
   │  └─[-] TURBO6                            TOTFAST .ASM      15,213  11-16-90
   │        ├─[+] ALC                       ▶ TOTIO2  .BAK      63,152  09-28-92
   │        ├─[ ] BGI                         TOTLIST .BAK      54,509  01-16-92
   │        ├─[ ] COMPRESS                    TOTMENU .BAK      55,045  01-23-92
   │        ├─[ ] DEM                       ▶ TOTMSG  .BAK      13,603  07-10-91
   │        ├─[+] DEMOS                     ▶ TOTWIN  .BAK      36,554  02-03-92
   │        ├─[ ] DOC                         README  .COM      11,297  02-09-91
   │        ├─[ ] DOCDEMOS                    OVERLAY .DOC       4,150  04-07-91
   │        ├─[ ] DRIVES                    ▶ TURBO   .DSK       1,927  02-05-92
   │        ├─[ ] INSCRIBE                  ▶ DEMBR4  .ERR         276  11-25-90
   │        ├─[ ] MARTIN                    ▶ DEMDR1  .ERR         198  11-30-90
   │        ├─[ ] PIE                         DEMLS1  .ERR         181  11-26-90
   │        ├─[ ] RTLV                        TOTDIR  .ERR         307  11-29-90
   │        ├─[+] SPREAD                   → ▶ TOTIO1  .ERR         329  11-29-90
   │        ├─[+] TOT                   ↓         TOTLIST .ERR         339  11-27-90 ↓

 F10=Actions  Shift+F9=Command Prompt                                      6:12p
```

**Figure 2-9:** You can select diverse files, and they are displayed in reverse video.

3. Choose **F**ile⇨**S**elect All.
4. Choose **F**ile⇨**D**elete.

**NOTE**

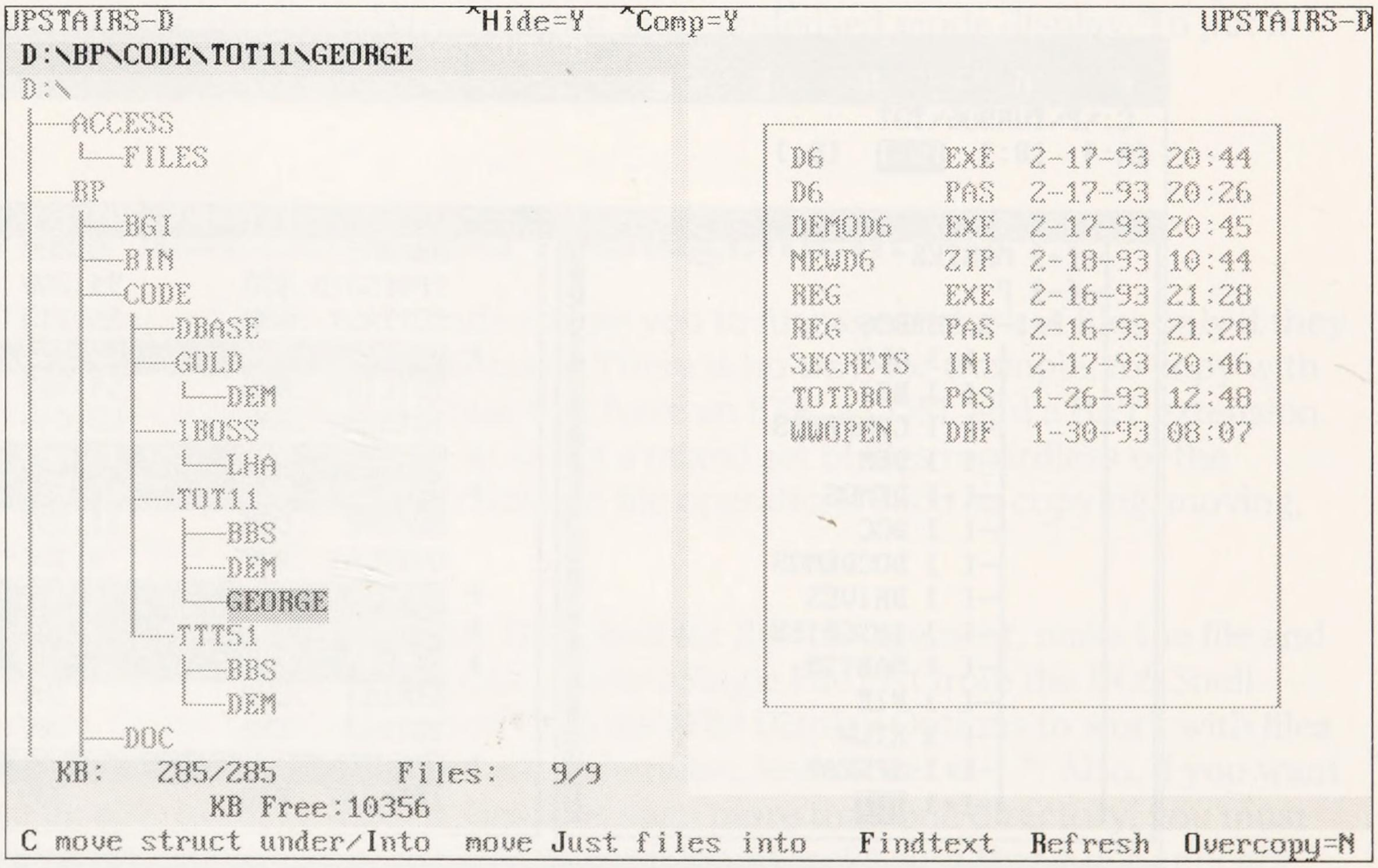
You can bypass the repetitive confirmation prompts by choosing **O**ptions⇨**C**onfirmation and unchecking the Delete field.

**DISK**

If you are using the DOS Shell for file management, I have good news for you. Bob's Shareware Collection disks include the excellent file manager CMFILER, from NovaSoft. CMFILER is a lightning-fast, full-screen utility that makes the DOS Shell look primitive.

CMFILER can display two directories side by side or display a directory tree in one window and a file list in the other (see Figure 2-10). In addition to enabling you to copy, move, delete, and rename files, you can browse, edit, and concatenate files. You can even add notes or comments to any file. CMFILER can print files or directory listings in the background, which enables you to continue with other file-management tasks and not wait for the print task to finish.

CMFILER also acts as a command center for executing programs, and you can optionally assign programs to the function keys. See Part V of this book for a detailed description of CMFILER.



```
UPSTAIRS-D                      ^Hide=Y  ^Comp=Y                  UPSTAIRS-D
 D:\BP\CODE\TOT11\GEORGE
 D:\
 |----ACCESS
 |   └---FILES
 |----BP
 |   |---BGI
 |   |---BIN
 |   |---CODE
 |   |   |---DBASE                       D6        EXE  2-17-93 20:44
 |   |   |---GOLD                        D6        PAS  2-17-93 20:26
 |   |   |   └---DEM                     DEMOD6    EXE  2-17-93 20:45
 |   |   |---IBOSS                       NEWD6     ZIP  2-18-93 10:44
 |   |   |   └---LHA                     REG       EXE  2-16-93 21:28
 |   |   |---TOT11                       REG       PAS  2-16-93 21:28
 |   |   |   |---BBS                     SECRETS   INI  2-17-93 20:46
 |   |   |   |---DEM                     TOTDBO    PAS  1-26-93 12:48
 |   |   |   └---GEORGE                  WWOPEN    DBF  1-30-93 08:07
 |   |   └---TTT51
 |   |       |---BBS
 |   |       └---DEM
 |   |---DOC
   KB:   285/285       Files:   9/9
           KB Free:10356
 C move struct under/Into  move Just files into   Findtext  Refresh  Overcopy=N
```

**Figure 2-10:** CMFILER can display a directory tree in one window and a directory list in the other.

## Task switching

The DOS Shell includes a task switcher that enables you to run more than one program at a time and then switch or jump from one active program to another. You may have both your word processor and your spreadsheet running at the same time, for example. Note that, unlike Windows, the DOS Shell does not allow one program to run in the background (to recalculate a spreadsheet, for example) while another application is being used. As soon as you switch from one application to another, the old application is suspended.

Although the DOS Shell task switcher has been criticized for being slow, it may well be much faster than having to repeatedly shut down one program and then launch another.

To run multiple programs, you must enable task switching by selecting **Options⇨Enable Task Swapper**. A dot appears next to the menu option when task switching is enabled. Many of the ways to launch programs and swap between them are the same as in Windows. This section shows some guidelines for how to perform the following main task-switching operations:

- Add programs to the menu
- Start programs
- Load programs
- Swap between programs
- Terminate programs

**Adding programs to the menu.** To add a program or submenu to the menu window (which is in the lower left quadrant of the display — refer to Figure 2-9), activate the window and select **File⇨New**. You then can create a new program entry by selecting Item or create a new submenu by selecting Group.

All the DOS Shell configuration details are stored in the file DOSSHELL.INI. If you have upgraded from DOS 5 to DOS 6, the setup procedure does not overwrite the existing DOSSHELL.INI file. The default options in the DOS 5 version of DOSSHELL.INI included a Disk Utilities menu that called the outdated DOS BACKUP and RESTORE programs. If you have not customized the DOS Shell and you want to use the new DOS 6 factory defaults (which call DEFRAG, MSBACKUP, and the other new programs), delete DOSSHELL.INI before installing DOS 6.

If you have already installed DOS 6, you can manually load an updated initialization file from Disk 1 of the DOS 6 installation disks. Be aware that any changes you already have made to customize the DOS Shell will be overwritten. Three files contain copies of DOSSHELL.INI: MONO.IN_, CGA.IN_, and EGA.IN_; these files are for monochrome, CGA, and EGA/VGA systems, respectively. Select the file that corresponds to your system's video display type. (Remember that you can use MSD to ascertain your display type if you don't know what it is.)

Because these files contain compressed data, they must be *expanded*, not copied, to the DOS directory. To expand the appropriate file (assumed to be EGA.IN_), insert Disk 1 into the drive (assumed to be drive A) and enter the following command:

```
a:expand a:ega.in_ c:\dos\dosshell.ini
```

The next time you run the DOS Shell, you see the new DOS 6 menu options.

**Starting programs.** You can start a program by using a variety of methods. The most obvious way is to select the item from the Program menu. Another way is to double-click the mouse on any program or batch file in the file list or highlight the program and press Enter. If you have *associated* a file extension with a specific program, you can select any file with that extension, and the associated program will be launched. You can also launch a program by selecting File⇨**R**un from the menu.

**Loading programs.** If you want to load a program but not start using it right away, hold down the Shift key before pressing Enter or clicking the left mouse button. The program is added to the Active Task List window, but the main DOS Shell window remains displayed.

**Swapping between programs.** You can return to the DOS Shell application from any program by pressing Ctrl-Esc, but be patient because it can take a few seconds. To use a different program, simply select it from the Active Task List. You can rotate from program to program by holding down the Alt key and pressing Tab. If you keep the Alt key pressed down, you can press Tab additional times to rotate through the list of programs. Release the Alt key when you reach the program you want. You can also assign a unique hotkey to any application when you first add the program to the menu, or later, by highlighting the menu item and selecting File⇨**P**roperties.

**Terminating programs.** The normal way to remove a program from the Active Task List is to swap to the program and quit the application in the normal manner. You can also terminate a program by highlighting the program in the Active Task List and pressing the Del key. Remember, however, that the application is terminated just as though you had turned the computer off; no files are saved, and the program does not follow its normal exit procedures. Avoid using Del, except when a program cannot be terminated in the normal manner.

## Summary

In this chapter, I have described some of the ways you can leverage maximum power from the most frequently used DOS commands. The important features you learned about in this chapter are shown in this list:

- The HELP command is packed with useful information about DOS commands and device drivers, and includes a search option for scanning for any text.
- You should use XCOPY in preference to COPY because, in most cases, it is faster and more versatile.
- The new MOVE command can move multiple files in a single operation, but, unlike COPY, a source filemask and a target path must always be specified. You can also use MOVE to rename directories.
- The new command DELTREE can erase an entire set of directories and files in one command, and it should be used with caution. DELTREE can delete hidden, system, and read-only files.
- Most users don't tap the power of DIR: By using the /S switch, you can search for files; by using the /A switch, you can list files that match a specific attribute; you can use the DIRCMD environment variable to customize DIR's default settings.
- The system diagnostic program MSD analyzes your system's hardware and can generate a comprehensive report that documents all aspects of the system's configuration. MSD can be used also to search for files.
- Although the DOS Shell is slow and a little clunky, you might consider using it for file viewing, task switching, and managing blocks of files that don't share a common filemask.
- Many excellent shareware programs included on Bob's Better-than-DOS Shareware Collection disks provide more power, functionality, and flexibility than do their DOS counterparts.

In Chapter 3, you learn which precautions you should take to avoid data loss and ways to recover from accidental file deletion, disk formatting, and virus infection.

## Chapter 3

# Protecting and Recovering That Irreplaceable Data

### *In This Chapter*

- The DOS 6 defensive arsenal
- Backup tips, traps, and truths
- "Must do" data-protection precautions
- Lost file and data recovery
- Deliberate data destruction
- Vital virus measures
- A lighter note

## The DOS 6 Defensive Arsenal

More and more businesses and livelihoods depend on the data stored in PCs. The loss of this data can be a real catastrophe. Fortunately, DOS 6 provides an arsenal of tools to help you protect against data loss and tools to help you recover if data loss occurs.

The following list summarizes the data-protection tools available in DOS 6:

MIRROR: Using the /PARTN switch, MIRROR saves a copy of the hard disk partition table to disk. This program is available on the DOS 6 Supplemental disk.

MSAV: The Microsoft Anti-Virus utility for identifying and removing (or cleaning) computer viruses. You can execute MSAV as a full-screen application or directly from the command line by using switches.

MSBACKUP: The Microsoft Backup utility for backing up *and* restoring files. The MSBACKUP full-screen utility does not support full execution directly from the command line.

MWAV: The Windows version of the Microsoft Anti-Virus utility. The core utility functions are the same as MSAV, but MWAV takes advantage of Windows' rich graphical environment.

MWBACKUP: The Windows version of the Microsoft Backup utility. The core features of both backup utilities are identical, and backups performed with MWBACKUP and MSBACKUP are compatible and interchangeable.

MWUNDEL: This Windows version of UNDELETE is used to recover accidentally deleted files.

UNDELETE: This tool provides mechanisms for enhancing the chances of recovering deleted files and provides tools to undelete files that have been deleted.

UNFORMAT: This tool recovers files and directories destroyed by an accidental format.

VSAFE: This tool loads the anti-virus TSR to monitor disk activity and protect against virus infection and damage.

Refer to Part IV, "The Complete DOS 6 Command Reference," for a full description of each of these commands.

When you install the Windows applications, SETUP automatically adds to the File Manager a Tools menu item with Backup and Anti-Virus menu options. If you install Windows after you install DOS 6, you can update File Manager with the new menu by running SETUP with the /E switch:

```
setup /e
```

Users of Norton Desktop for Windows (NDW) 2.0 or earlier may be irritated by the appearance of two (count 'em) Tools menus on the desktop. One is the NDW standard Tools menu, and the other one is the Tools menu created by DOS 6. NDW reads the [ADDONS] section of WINFILE.INI and, like File Manager, uses this information to create additional menus. You can instruct NDW to ignore the menu add-ons (and therefore eliminate the extra Tools menu) by adding the following entry to the [DEFAULTS] section of NDW.INI:

```
MaxWinFileExtensions=0
```

DOS 6 SETUP also creates a program group labeled Microsoft Tools that provides quick access to the three Windows-based applications: Anti-Virus, Backup, and Undelete. This group is defined in the file WNTOOLS.GRP in the DOS directory. NDW users should manually create a Microsoft Tools group by selecting the main menu options File⇨New⇨Group and using the browse facility to select the WNTOOLS.GRP file from the DOS directory.

When you use the Anti-Virus, Undelete, or Backup utilities from a shared network drive, you may find that another network user has reconfigured the tools. All three data-protection utilities access INI files — namely, MSAV.INI, MWAV.INI, UNDELETE.INI, and MSBACKUP.INI — to determine their default characteristics.

By creating an environment variable called MSDOSDATA, you can instruct these utilities to access the INI file in a specific directory. Just set the environment variable MSDOSDATA equal to the specific directory in which the personal INI file is located. You should add the following command, for example, to AUTOEXEC.BAT to force DOS to load the INI files from the C:\INIFILES directory:

```
msdosdata=c:\inifiles
```

When you customize a utility's configuration, the changes are stored in your personal INI file.

## Backup Tips, Traps, and Truths

The best way to deal with data loss is to be prepared for it, which leads us to the first truth:

**Truth 1: At one time or another, an important file will be destroyed.**

Although you may be able to use a utility program to recover a deleted file, in many cases the file is destroyed and cannot be recovered. If you accidentally overwrite a file by using the same name to copy or move another file into the same directory, for example, the chances that you can recover the overwritten file are slim to none. Your only option is to restore a copy of the file from a backup disk. Obviously, you can restore the file from a backup only if you *have* a backup.

It is all too easy to destroy a valuable file by using the COPY or MOVE command. One of the many valuable features of the shareware program PCOPY (part of Patriquin's Hard Disk Utilities) is that it prompts you before overwriting a file during a copy or move operation. You can, of course, instruct PCOPY to overwrite files automatically, but by default (thankfully), it doesn't.

Even with first-class data-recovery tools at your disposal, up-to-date file backups are essential.

## Using the DOS 6 backup software

With the release of version 6, DOS backup tools suddenly got much better, and it's about time. Thankfully, the much-criticized BACKUP and RESTORE programs have been replaced by an all-new backup utility, which comes in two flavors: MSBACKUP for character-mode DOS and MWBACKUP for Windows.

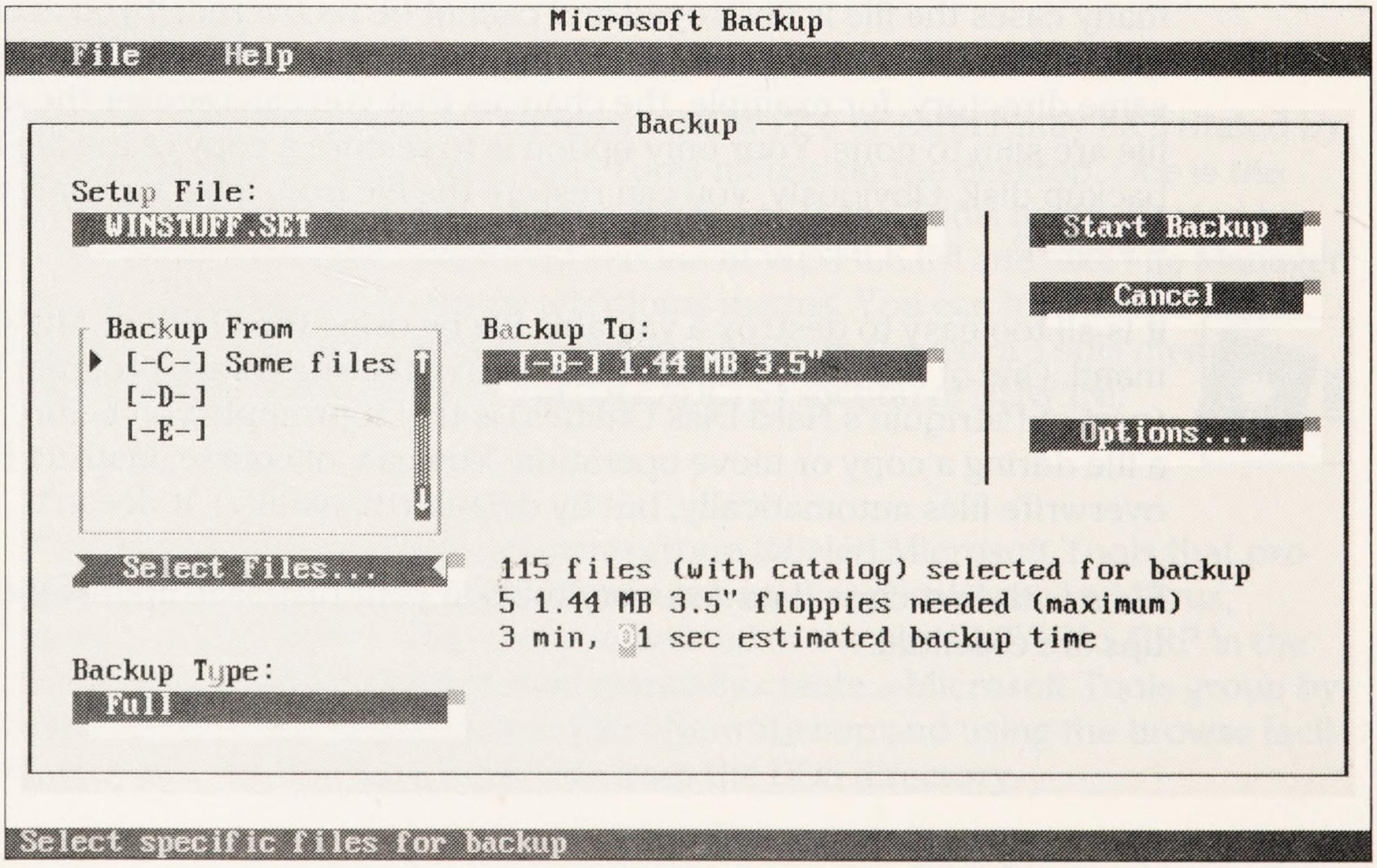
Although the RESTORE program ships with DOS 6, it is provided only so that you can restore files made with DOS's old BACKUP program.

### Comparing MSBACKUP and MWBACKUP

Other than their obvious cosmetic differences, the two backup utilities MSBACKUP and MWBACKUP are very similar. They feature the same backup options, use similar terminology and menus, and offer similar performance. What's more, backups created with one product can be restored with the other. Figures 3-1 and 3-2 show the similarities between the two products highlighted in the Backup dialog boxes.

Use whichever utility you are most comfortable with.

The backup programs can back up files only to a standard DOS device, such as a floppy disk, hard disk, network drive, or removable drive (a Bernoulli box, for example). A major drawback is that the programs cannot back up files to tape devices. I anticipate that this feature will be added in a future DOS release.



Setup File: WINSTUFF.SET

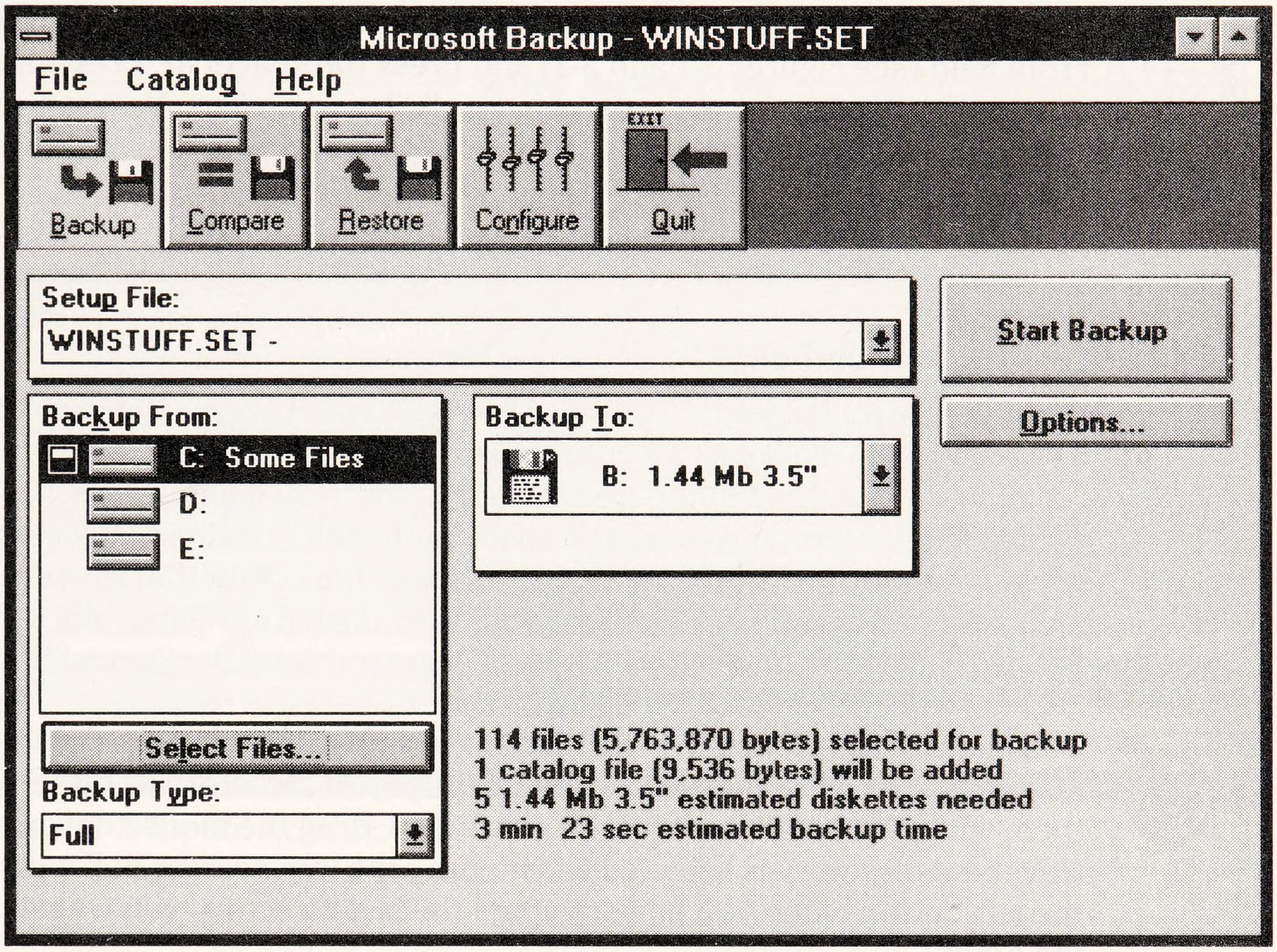
Backup From: ▸ [-C-] Some files
[-D-]
[-E-]

Backup To: [-B-] 1.44 MB 3.5"

115 files (with catalog) selected for backup
5 1.44 MB 3.5" floppies needed (maximum)
3 min, 31 sec estimated backup time

Backup Type: Full

**Figure 3-1:** The Backup dialog box from MSBACKUP, the Text Mode backup utility.



Microsoft Backup - WINSTUFF.SET

Setup File: WINSTUFF.SET -

Backup From:
C: Some Files
D:
E:

Backup To: B: 1.44 Mb 3.5"

Backup Type: Full

114 files (5,763,870 bytes) selected for backup
1 catalog file (9,536 bytes) will be added
5 1.44 Mb 3.5" estimated diskettes needed
3 min 23 sec estimated backup time

**Figure 3-2:** The Backup dialog box from MWBACKUP, the Windows backup utility.

The first time you run the backup software, you must go through a configuration process. During the configuration, the backup program backs up a little more than one floppy disk full of files and then runs a comparison to verify that the files were backed up successfully. Be prepared to have two disks and don't use disks that have valuable data on them.

If your system has more than one floppy drive, you can choose which drive is used for the minibackup. Be sure to select the drive you normally use for backups — normally, the highest capacity drive.

## Selecting files to back up

As bizarre as it may sound, most people have difficulty selecting which files to back up. If you think you just execute the program and select the backup button, think again!

The list of files you select to back up is stored in a SETUP file that has a SET extension. By default, the backup program loads the file DEFAULT.SET.

You must have at least one file tagged for backup before you can run a backup. If no files are tagged for backup, the Start Backup button is grayed.

A quick way to select all the files on a drive is to highlight a drive in the Backup From field (see Figures 3-1 and 3-2) and press the spacebar. Pressing the spacebar again toggles the setting from all files to no files. You can select files from multiple drives in a single backup.

To select some but not all files from a drive, you can choose from two distinct techniques: the point-and-click method and include/exclude clauses.

### Using point and click

From the Select Backup Files dialog box, you can traverse the directory tree and select files. To select an entire directory, press the spacebar while the directory is highlighted in the tree. To select individual files, press Tab to move to the file list window and press the spacebar to select individual files. You can press the spacebar a second time to deselect files. Note that to view all files, including hidden and system files, you must uncheck the Exclude options in the Special Selections dialog box, which is accessed from the Special button.

You can use the mouse to select multiple contiguous branches of the tree in one operation. First double-click on the topmost directory to be selected and then, while you keep the left button pressed, drag the mouse downward. Each traversed directory is selected as you drag the mouse over it. Finally, release the left button when you have reached the last directory you want to select in the block.

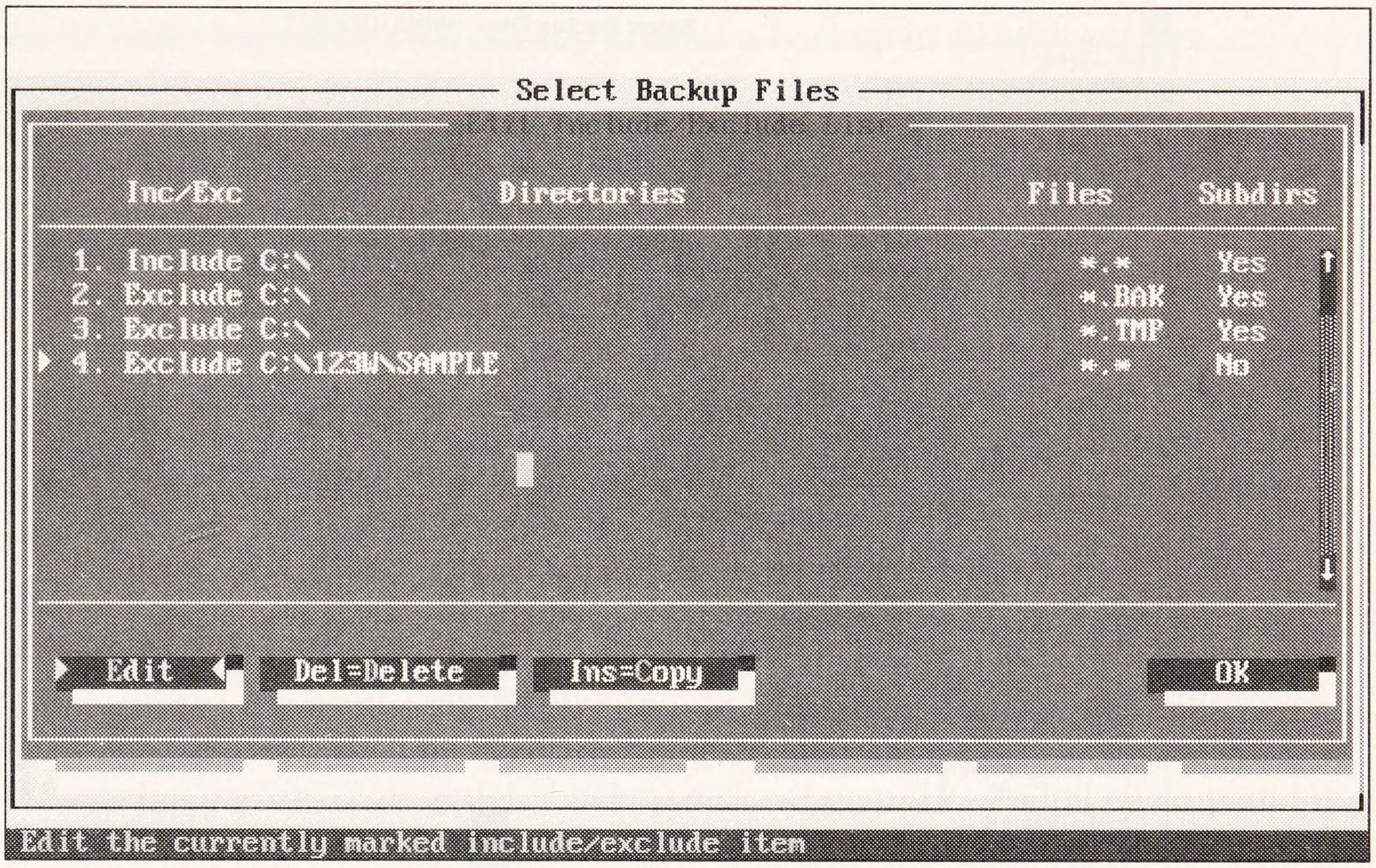
Selected files are identified by a check mark ( ). When a file is selected individually but is excluded because of an overriding exclusion clause or because it doesn't match the backup type, a dot (·) is displayed. Files that display a dot are not backed up.

Although the two backup programs are virtually identical, file selection is one area in which the Windows program has the edge. The use of the + and - keys expands and collapses the directory tree and makes it easy to select an entire drive by collapsing the root directory. You cannot collapse the tree in MSBACKUP.

### Using include/exclude clauses

My preferred method of file selection is to include and exclude directories explicitly by using the Edit Include/Exclude List dialog box (or the Include/Exclude Files dialog box in the Windows version). In the DOS backup program, you should click on either the **I**nclude or E**x**clude button and then immediately select the **E**dit Include/Exclude List button to bypass the next dialog box (which doesn't appear in the Windows version). Figure 3-3 illustrates this dialog box.

By specifying filemasks and directories, you can create a list of included and excluded files. You should note that the backup programs read the list in top-down order. If two conditions conflict (one clause says to include some files,



```
    Inc/Exc              Directories                     Files    Subdirs

 1. Include C:\                                          *.*      Yes
 2. Exclude C:\                                          *.BAK    Yes
 3. Exclude C:\                                          *.TMP    Yes
 4. Exclude C:\123W\SAMPLE                               *.*      No
```

```
Edit the currently marked include/exclude item
```

**Figure 3-3:** The Edit Include/Exclude List dialog box from MSBACKUP.

and another clause says to exclude the same files, for example), the clause that is lower down the list is honored. Consider the following clause list:

```
INCLUDE D:\                                Yes *.*
EXCLUDE D:\ACCESS                          Yes *.*
EXCLUDE D:\................................Yes *.BAK
```

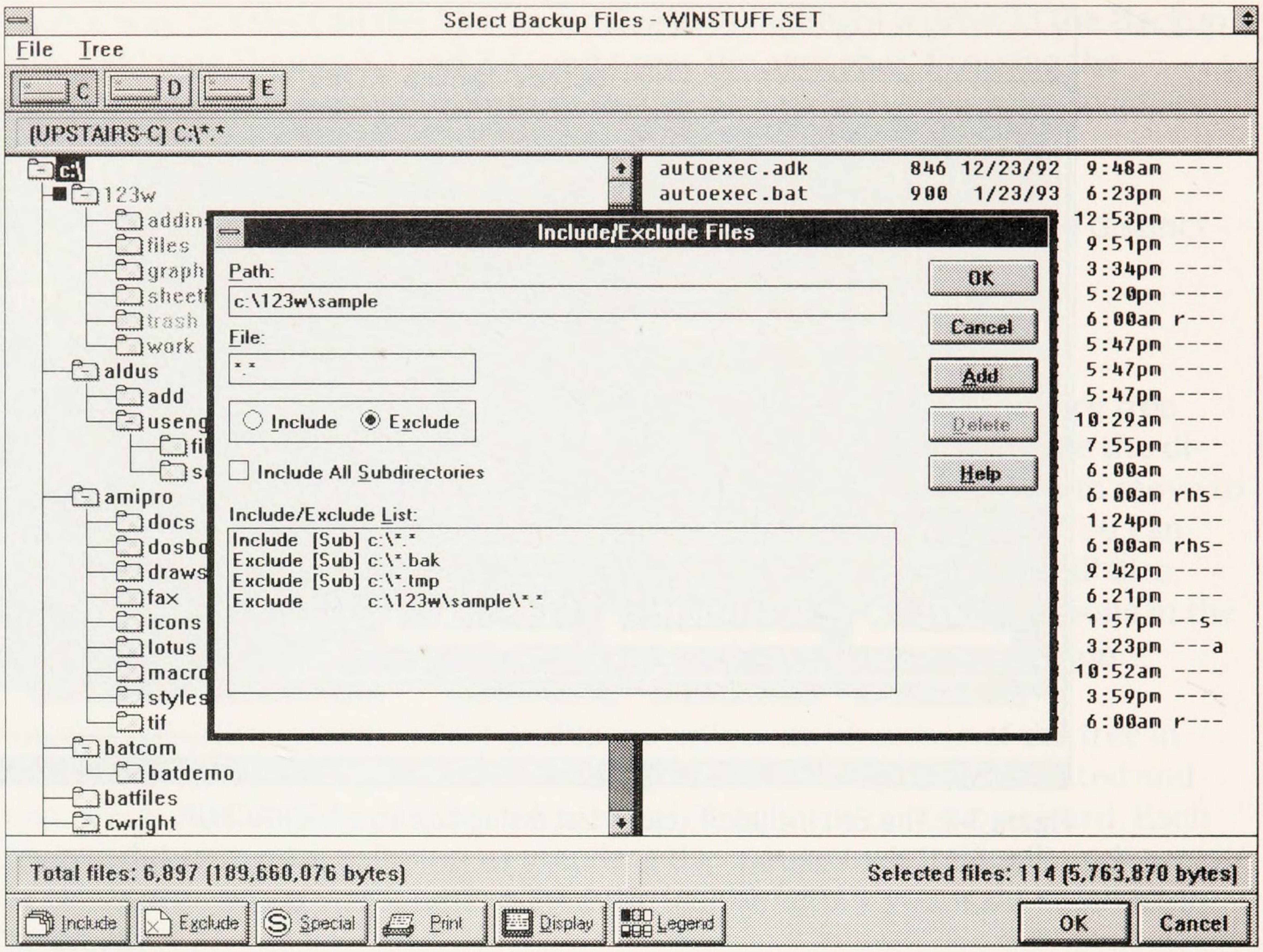
The preceding clause list includes all files in drive D except files in the \ACCESS subdirectory (and its subdirectories) and any files with a BAK extension.

If the order of the list is changed, however, the meaning is very different. Consider the following list:

```
EXCLUDE D:\ACCESS                          Yes *.*
EXCLUDE D:\................................Yes *.BAK
INCLUDE D:\                                Yes *.*
```

Because the last statement in the list includes all files on drive D, the two preceding EXCLUDE items are overwritten (ignored). This list results in the inclusion of all drive D files.

Figure 3-4 shows the Include/Exclude Files dialog box in the Windows-based backup program. I can't count the number of times I have seen a user define an include or exclude clause and then promptly lose it by pressing Enter. Only the items listed in the lower dialog box field titled Include/Exclude List are applied.


Select Backup Files - WINSTUFF.SET
File
Tree
C
D
E
(UPSTAIRS-C) C:\*.*
c:\
123w
addin
files
graph
sheet
trash
work
aldus
add
useng
fi
s
amipro
docs
dosbo
draws
fax
icons
lotus
macro
styles
tif
batcom
batdemo
batfiles
cwright
Include/Exclude Files
OK
Cancel
Add
Delete
Help
Include
Exclude
Special
Print
Display
Legend
OK
Cancel

```
autoexec.adk       846 12/23/92   9:48am ----
autoexec.bat       900  1/23/93   6:23pm ----
```

Path: c:\123w\sample

File: *.*

○ Include ● Exclude

☐ Include All Subdirectories

Include/Exclude List:

```
Include  [Sub] c:\*.*
Exclude  [Sub] c:\*.bak
Exclude  [Sub] c:\*.tmp
Exclude        c:\123w\sample\*.*
```

Total files: 6,897 (189,660,076 bytes)

Selected files: 114 (5,763,870 bytes)

**Figure 3-4**: The Include/Exclude Files dialog box from MWBACKUP.

After you have entered a new clause in the top fields, you must click on the Add button — pressing Enter simply selects the OK button, which ignores the last entry and removes the dialog box. Crazy.

When you return to the main backup window, a summary of the number of selected files is displayed in the lower right quadrant of the window (refer to Figure 3-1).

Having gone through the pain of selecting the files you want, you can save the selection in a SET file. The next time you want to back up the same collection of files, just load the SET file from the File menu. I keep a variety of SET files to handle all my backup needs: one saves all my programming source files, for example, and another one saves all the files used in the creation of this book.

## Selecting the backup type

In addition to the selection of individual files, the backup programs support these three types of backups:

- Full
- Incremental
- Differential

As its name suggests, a *full* backup creates a backup of all *selected* files.

An *incremental* backup makes a backup of all files that have been changed (or added) since the last full or incremental backup. This backup type references the file archive bit to determine whether the file should be included in the backup. If the archive bit is set, the file is backed up; otherwise, the file is not backed up. When the file has been backed up, the archive bit is turned off.

**CAUTION**

Do not use the same backup disks for saving every incremental backup. You can restore the complete file system from only the previous full backup and *each* subsequent incremental backup. Always use fresh disks for each incremental backup, until the next full backup is taken.

A *differential* backup makes a backup of all files that have been changed since the last full backup. Like an incremental backup, a differential backup uses the archive bit to determine which files should be backed up. The main difference between these two backup types is that the archive bit is not turned off after a differential backup. A complete backup disk set comprises a full backup and the last differential backup.

**CAUTION**

Between full backups, you should perform either incremental backups or differential backups. Do not mix the two backup types. The task of restoring a complete file set is significantly complicated when you have used both partial backup types.

You can change the backup type by using the Backup Type field in the main Backup window.

## Special file-selection options

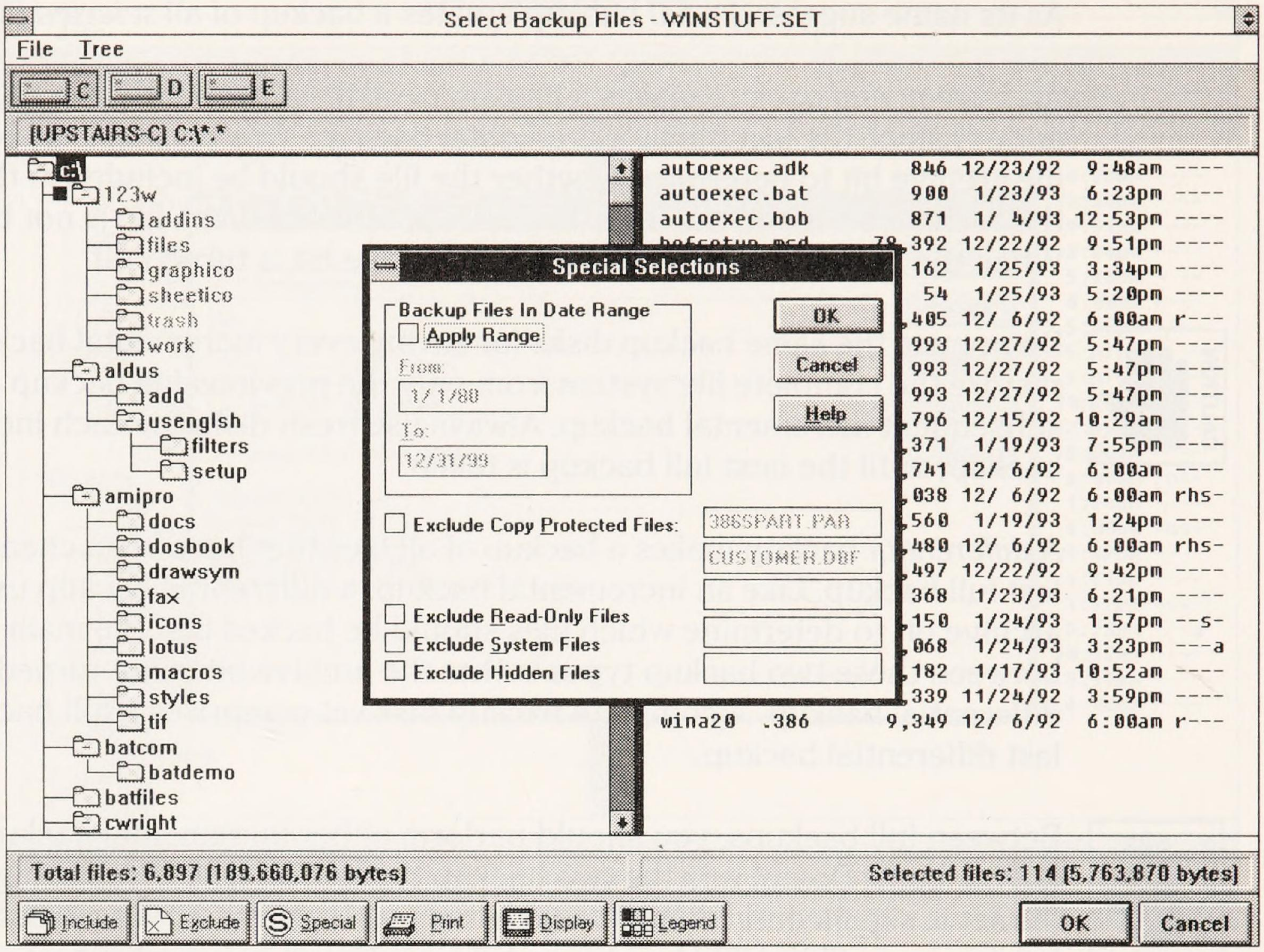
In addition to individual file selection and the backup type, a set of special options narrows the file selection. You can confine the backup to include only files with a date that falls within a specific range or exclude files based on their read-only, hidden, and system attributes.

The Special Selections dialog box is displayed when you select the Special button in the Select Backup Files window (see Figure 3-5).

The dialog box is self-explanatory except for the five fields in the lower right quadrant. These fields are designed for specifying copy-protected files that should not be backed up. They enable you to specify as many as five files that are not backed up from any directory if the Exclude Copy Protected Files field is checked.

**GEM**

Thankfully, copy-protected files are less prevalent nowadays. One file I always want to avoid backing up is the Windows permanent swap file 386SPART.PAR. This file is invaluable for increasing the capacity of Windows, but it contains no data that has to be backed up. A sure-fire way to always exclude this file from a backup is to identify it as a copy-protected file in the Special Selections dialog box.


Select Backup Files - WINSTUFF.SET
File
Tree
C
D
E
c:\
123w
addins
files
graphico
sheetico
trash
work
aldus
add
usenglsh
filters
setup
amipro
docs
dosbook
drawsym
fax
icons
lotus
macros
styles
tif
batcom
batdemo
batfiles
cwright
Special Selections
OK
Cancel
Help
Include
Exclude
Special
Print
Display
Legend
OK
Cancel

[UPSTAIRS-C] C:\*.*

Backup Files In Date Range

☐ Apply Range

From: 1/ 1/80

To: 12/31/99

☐ Exclude Copy Protected Files: 386SPART.PAR, CUSTOMER.DBF

☐ Exclude Read-Only Files

☐ Exclude System Files

☐ Exclude Hidden Files

```
autoexec.adk          846 12/23/92  9:48am ----
autoexec.bat          900  1/23/93  6:23pm ----
autoexec.bob          871  1/ 4/93 12:53pm ----
bofsetup.msd       78,392 12/22/92  9:51pm ----
                      162  1/25/93  3:34pm ----
                       54  1/25/93  5:20pm ----
                     ,405 12/ 6/92  6:00am r---
                      993 12/27/92  5:47pm ----
                      993 12/27/92  5:47pm ----
                      993 12/27/92  5:47pm ----
                     ,796 12/15/92 10:29am ----
                     ,371  1/19/93  7:55pm ----
                     ,741 12/ 6/92  6:00am ----
                     ,038 12/ 6/92  6:00am rhs-
                     ,560  1/19/93  1:24pm ----
                     ,480 12/ 6/92  6:00am rhs-
                     ,497 12/22/92  9:42pm ----
                      368  1/23/93  6:21pm ----
                     ,150  1/24/93  1:57pm --s-
                     ,068  1/24/93  3:23pm ---a
                     ,182  1/17/93 10:52am ----
                     ,339 11/24/92  3:59pm ----
wina20   .386       9,349 12/ 6/92  6:00am r---
```

Total files: 6,897 (189,660,076 bytes)

Selected files: 114 (5,763,870 bytes)

**Figure 3-5:** The Special Selections dialog box from MWBACKUP.

## Make sure that your backup is good

**Truth 2: Many people don't discover that their backups are useless until they try to recover an important file.**

Both backups provide a Compare option for verifying that the backed-up data is the same as the source data on the hard disk. For the first few backups, run the Compare option to verify that you are making good backup copies. This option takes time, but there is no point in wasting time backing up files if you cannot restore them later.

As an acid test, you should consider restoring a single file from a backup to make sure that, in a real emergency, you can do likewise. Pick a single file to restore as a test and rename that file on the hard disk. Then try to restore the file from the backup.

## Optimizing backup speed

In my book, the faster the backup, the better. The longer a backup takes, the less frequently I make them. Follow these steps to maximize backup speed:

- Use high-density floppy disks.
- Turn off data verification (from the Options menu). There is, of course, a trade-off between performance and safety. Use the Compare option a number of times to occasionally spot-check that the backup is sound. If you are using new disks for a backup, turn on data verification for the first backup.
- Turn on data compression. It takes less time to compress the data and write fewer bytes to the disk than it does to blindly save the uncompressed file to disk.
- Turn off error correction. Again, there is a trade-off between speed and safety here. By storing error-correction data on the backup disk, the chances of restoring files from the disk (after the disk has been damaged) are greatly enhanced. The error-correction information requires approximately 10 percent additional disk space.
- In the Windows environment, keep the other active applications to a minimum. The more applications asking for a slice of the CPU's attention, the longer the backup takes.

TECHNICAL STUFF

Both backup programs use a special Error Correction Code (ECC) data-storage technique, pioneered by Symantec. This technique enhances your chances of restoring data from a damaged disk. This damage can occur when you make the backup with the Data Verification option disabled and the disk has some bad sectors, or when the disk becomes physically damaged after the backup is made.

The ECC system involves storing a group of sectors to the end cylinder of the disk. When you are restoring data, you can use the ECC section to rebuild as many as four damaged sectors.

CAUTION

When you use MWBACKUP, make sure that you do not run another application that accesses the floppy drives — data can be destroyed. This restriction applies to all floppy drives, not just to the one being used for the backup.

## Files created and used by the backup

A variety of files are created and accessed by the backup programs. The following list shows the main files, which are discussed in this section:

- SET
- SLT
- Catalog

- CAT
- 00*n*
- INI
- LOG

**SET files.** SET files store a full definition of a backup, including which drives to back up, which files to back up, which target drive to back up to, the backup type (full or incremental, for example), and the special options. SET files are stored in the DOS directory and are plain ASCII files. You can readily print and review them by using the File⇨Print menu option.

**SLT files.** An SLT file holds the list of directories and files that were selected for backup using the keyboard and mouse (rather than an include/exclude list). The file uses the same eight-character name as the SET file, but with an SLT extension.

**Catalog files.** A catalog file is created for every backup. The catalog file is binary and stores complete information about the structure of the drive (or drives) that was backed up, in addition to a complete list of all backed-up files. One copy of the file is stored in the DOS directory, and another copy is stored at the end of the backup data on the backup disks.

A simple convention is used to build the catalog filename. The eight-character filename details the backup source and date by using the following conventions:

| | |
|---|---|
| Character 1 | The letter of the first drive that had files backed up. |
| Character 2 | The letter of the last drive that had files backed up. |
| Characters 3–7 | The date when the backup was made using the format YMMDD, where Y is the last digit of the year when the backup was made (4 for 1994, for example). |
| Character 8 | A letter that identifies the order of the backup. This character really is used to ensure that every catalog has a unique name. If the backup program sees that a catalog already exists with the same first seven digits (and the same extension), the next available letter of the alphabet is used in character position 8. |

NOTE

If the Keep Old Catalogs option is turned off, the eighth character always is A, and any existing catalog with the same name is overwritten.

The catalog filename extension indicates the type of backup and is FUL, INC, or DIF.

**CAT files.** A master catalog (CAT) file is created for every backup set. The filename comprises the first eight characters of the SET filename and has a CAT extension. This (ASCII) file contains a record of when the last full backup was made and a chronological history of every subsequent incremental or differential backup. This file is accessed during the restore process. Every time a new full backup is taken, the file is cleared and the new backup is recorded as the first entry.

**00*n*.** The files created on the backup disks with a 00*n* extension format are the files in which the backed-up data is stored. The first backup disk has the extension 001; the second, 002; and so on. The eight-character filename is built by using the same conventions as the catalog file. The date and time stamp on the file indicates the date and time the backup was made. You can use old backup disks that were formatted by the BACKUP program as general-purpose DOS disks.

GEM

The volume label of a backup disk provides more information about the backup. The first part of the volume label is the name of the setup file (without the SET extension) that was used to define the backup options. The last three characters of the volume label indicate the backup type and are FUL, INC, or DIF.

**INI files.** MSBACKUP creates the binary (noneditable) file MSBACKUP.INI, which defines the primary backup configuration and system characteristics (DMA transfer type and installed drives, for example). If this file is not present, you must configure MSBACKUP before you can use it.

**LOG files.** The temporary file MSBACKUP.LOG contains (in binary format) the directory structure of all the drives that were logged in the last MSBACKUP session. This file is created when MSBACKUP displays the message that it is reading disk information.

GEM

Any floppy disk used for a backup can be used later as a standard DOS floppy disk for copying files and other tasks. If you try later to use the old disk in a new backup, the backup program presents a warning message that the disk was used in a previous backup. This message is displayed even when you have deleted the old backup files and changed the volume label.

You may wonder how the backup program knows that the disk has been used in an earlier backup. The backup program records the backup details (including the SET name, the date and time of the backup, and the floppy disk number) behind the scenes in part of the boot sector. This area is erased only when you format the disk. Whenever you insert a new disk, BACKUP checks the boot sector to see whether the disk has already been backed up to. This defensive mechanism stops you from inadvertently inserting the wrong floppy disk during a backup session.

## Restoring files from a backup

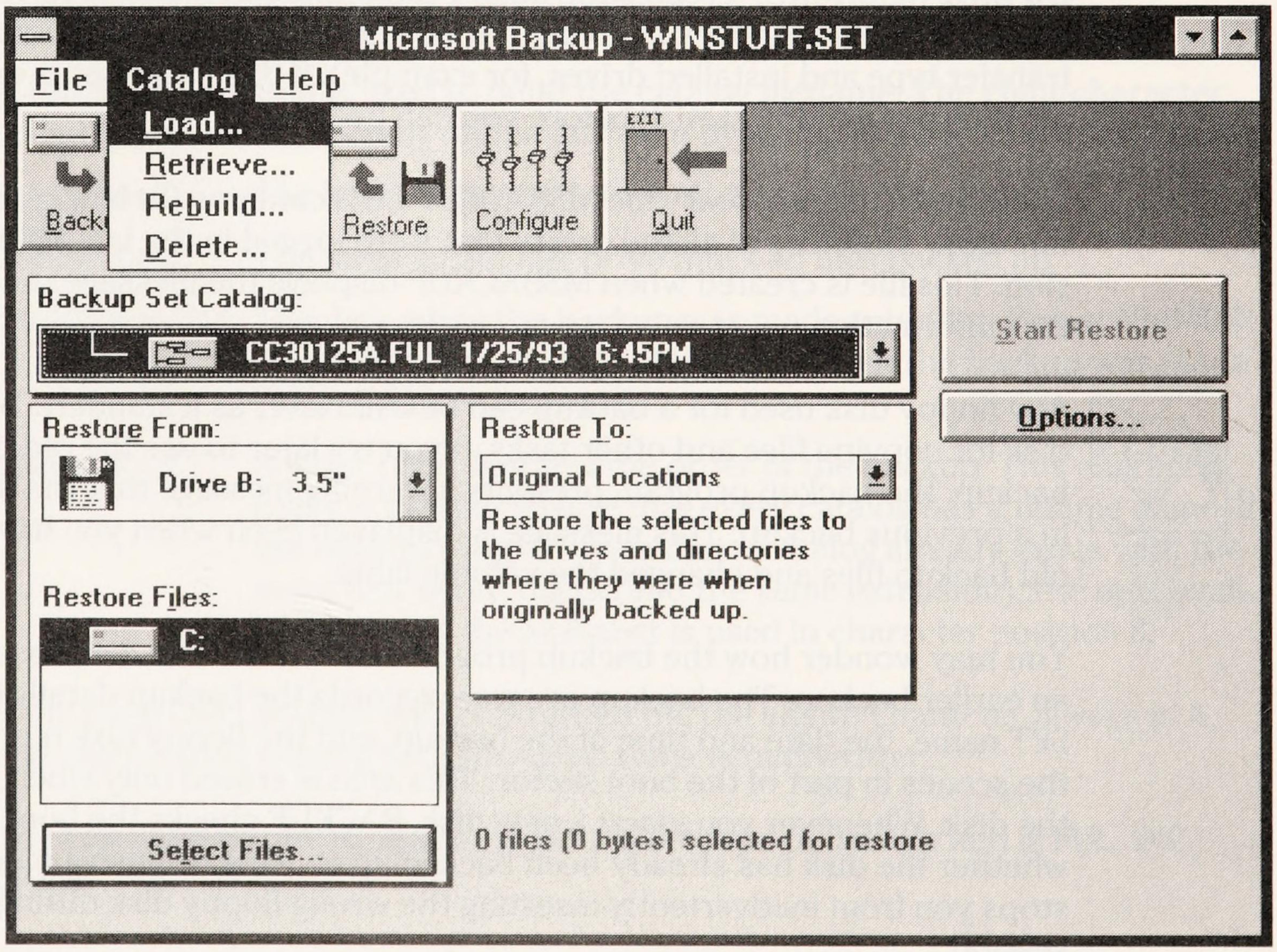
**Truth 3: A backup is useless if you can't restore any files from it.**

You can restore files only after the backup program has loaded the backup's corresponding catalog file. When the backup is made, one catalog file is saved in the DOS directory, and the other is saved on the last backup disk.

Figure 3-6 shows the catalog-selection options presented in the Select Catalog dialog box. You can display this dialog box by selecting the Catalog button in the Restore window. Although MWBACKUP does not provide a catalog dialog box, you can select the same options from the Restore main menu (see Figure 3-7).

You can access a catalog in one of these three ways:

| | |
|---|---|
| Load | The CAT file is retrieved from the hard disk. |
| Retrieve | The CAT file is restored from the last floppy disk in the backup set. |
| Rebuild | The backup program examines the contents of every floppy disk in the backup series and re-creates the CAT file. |


Microsoft Backup - WINSTUFF.SET
File
Catalog
Help
Load...
Retrieve...
Rebuild...
Delete...
Back
Restore
Configure
EXIT
Quit
Start Restore
Options...
Select Files...

Backup Set Catalog: CC30125A.FUL 1/25/93 6:45PM

Restore From: Drive B: 3.5"

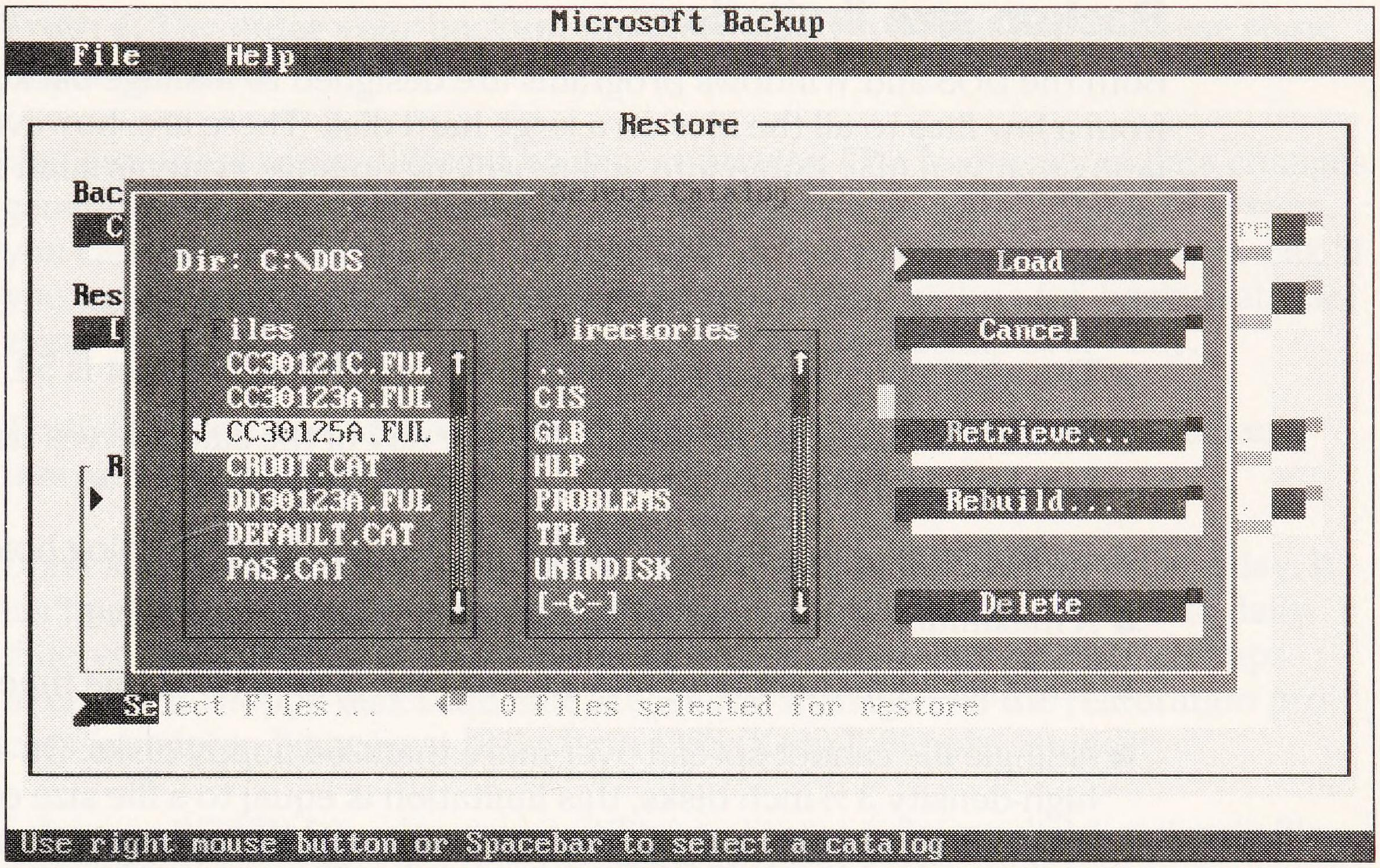
Restore To: Original Locations

Restore the selected files to the drives and directories where they were when originally backed up.

Restore Files: C:

0 files (0 bytes) selected for restore

**Figure 3-6:** The Select Catalog dialog box from MWBACKUP.



```
Dir: C:\DOS

Files             Directories
CC30121C.FUL      ..
CC30123A.FUL      CIS
√ CC30125A.FUL    GLB
CROOT.CAT         HLP
DD30123A.FUL      PROBLEMS
DEFAULT.CAT       TPL
PAS.CAT           UNINDISK
                  [-C-]

Select Files...   0 files selected for restore

Use right mouse button or Spacebar to select a catalog
```

**Figure 3-7:** The catalog menu options in MSBACKUP.

Normally, you should try to load the catalog. If you can't find the CAT file, try to retrieve the catalog from the backup set. If the file is damaged or you can't find the last disk in the backup, rebuild the catalog from the disks you have.

After loading the catalog, you choose the files to be restored by using the same techniques you used to choose files to be backed up (discussed earlier in this chapter), except that you cannot use include and exclude lists. When you choose the Select Files button from the Restore menu, the directory tree reflects the directory structure of the disk when the files were backed up, and it can differ from the current directory structure.

If you have not selected any files, the Start Restore button is grayed and inactive.

By default, the backup program does not restore files that are read-only, hidden, or system. If you back up all the files on drive C, format the drive, and then use default settings to restore the files, for example, your drive will not be bootable because all the read-only, hidden, and system files will be missing. Select the Special button from the Select Restore Files dialog box and uncheck the exclude fields to ensure that all files are restored regardless of their attribute values.

### Backup size limitations

Both the DOS and Windows programs are designed to manage backups ranging from a few files to all the files on a large hard disk. There are, however, some limits that you may bump into when backing up huge systems such as a file server.

This list shows some of the backup program's operating limits:

- The maximum number of entries in an include/exclude list is 50.
- There is a maximum of 26 backups per day of the same drives using the same backup type.
- A maximum of 49 setup (SET) files can be listed in the dialog boxes.
- A maximum of 1,023 directories can be logged for any drive.
- The maximum number of files that can be backed up at one time is 65,535.
- A single file cannot spread over more than 254 floppy disks. On 1.44MB high-density 3 ½-inch disks, this limitation is equal to a file size of 730MB when maximum compression is used. Using 360K floppies with no compression, however, drops the maximum file size to a mere 90MB.

## Backing up DoubleSpace drives

All the files and directories on a DoubleSpace drive are stored in a single (humongous) hidden file on the host drive.

You could, in theory, back up the DoubleSpace files in one of two ways. You could back up the host drive and be sure to include hidden files in the backup. The drawback to this technique is that all the DoubleSpace files are stored in a single backup file. If, for some reason, this large file cannot be restored because of a minor data error, you could not restore any of the DoubleSpace files hidden within. You will have put all your eggs in one huge, fragile basket.

A better way is to back up the files on the DoubleSpace drive individually, like any other standard drive. Then, if a data error occurs, it probably will affect only one or two files.

## Developing a backup strategy

No discussion of backup tools is complete without a discussion of backup strategy.

**Truth 4: The older your backup, the more likely it is that you will lose some important data.**

You can adopt many different backup strategies. The frequency of data changes generally dictates how frequently you should back up. On a little-used system, you can do a full backup every month and an incremental backup every week. If you use your system on a full-time basis, you should make a full backup at least every week and make an incremental or differential backup every day.

**Truth 5: If you don't follow a strict backup regime, you will make backups less and less frequently.**

Don't set unrealistic backup goals, such as making a full backup twice a day. It isn't necessary. I prefer to make a full backup twice a week and then do daily differential backups. I prefer differential backups over incremental backups because the floppy disk sequence is easier to manage and the restoration process is simpler. I'm a great believer in KISS (Keep It Simple, Stupid).

DISK

I also use PCOPY (the shareware utility) to keep, on floppy disks, a simple file copy of important files (such as the DOC file for this chapter!).

**Truth 6: One backup copy is not enough.**

Imagine that you realize you edited the wrong worksheet last week and updated the master budget files rather than the monthly forecast files. You have to get the original budget file back, so you try to restore the original file from a backup. The only problem is that you took the last backup after you made the changes, so the backup contains the modified budget file as the one on the system. This scenario occurs more often than you can imagine.

The solution is to keep multiple backup sets. You may keep four old sets of backups, for example, one for each of the past four weeks. Every time you make a full backup, you overwrite the oldest backup disk set.

**Truth 7: If you keep your backup floppy disks near your computer, you are courting disaster.**

Always, always keep backup files in a different location from your computer, and preferably off-site. If your office (or home) gets damaged by fire or water, your backup will be destroyed with your computer.

CAUTION

If you have classified or confidential information on your computer, guard your backup floppy disks with the same fervor as you guard your system. There is little point in keeping your system secure when snoopers can restore on their system your files from the backup. I keep a complete backup in a safety-deposit box at the bank. Not only is it secure, but also the chances of the bank going up in flames at the same time as my office is remote enough for me.

**Truth 8: If making backups isn't convenient, you probably won't make them.**

Backing up a 500MB disk drive to floppy disks is downright impractical. If you have a large hard drive, you should set up an alternative to floppy disks.

If you are a network user, for example, you may be able to back up your local drives to the file server. As a bonus, the network administrator can keep back copies of all server volumes on tape.

If you are not connected to a network, consider installing a removable drive device, such as a Bernoulli drive.

More and more power users are installing tape-backup devices. Larger systems can hold 250 or 500 megabytes on a matchbox-size cassette. Unfortunately, the DOS backup programs do not (yet) support tape devices. If you are a heavy PC user with no other cost-effective alternative, you should consider installing a tape unit and buying a third-party backup program that supports tape drives. It may be the best $300 you spend.

## "Must Do" Data-Protection Precautions

No one takes a backup every 15 minutes. By implementing a few sound practices, you can put in place a defensive mechanism that will help you to survive unexpected system crashes, accidental file deletions, and virus attacks.

This section discusses some of the main precautions you should consider implementing. Don't forget, however, that there is no substitute for frequent data backups.

### Guarding against accidental file deletion

The most common cause of data loss is not by a virus attack or malice, or even by hardware failure: It is by you and me — human error. Put simply, we delete things by mistake.

DOS 6 provides the UNDELETE program to help protect against data loss caused by accidental file deletion. UNDELETE provides both data protection and data recovery. You can load it as a TSR that consumes about 14K of RAM and configure it to use one of the following two primary methods of deletion protection:

- Delete sentry
- Delete tracker

In the *delete sentry* method, a hidden directory named \SENTRY is used to store copies of deleted files. The TSR portion of UNDELETE monitors all file-deletion activity and substitutes a file move operation in place of the file deletion. This method provides the most secure form of protection because you can undelete the file just by moving it from the deleted directory back to the original directory. The problem with delete sentry is that it can hog disk space. After all, you don't recover any disk space when you delete a file because the file isn't really deleted.

The *delete tracker* method uses a hidden file named PCTRACKR.DEL in the root directory of the monitored drive to store vital data recovery information. This file contains the name of the deleted file and a list of all the clusters in which the file's data was stored. This method doesn't offer as much protection as the delete sentry method because other files can still overwrite the clusters. The primary advantage is that much less disk space is consumed. What's more, the chances of file recovery are good if there has been limited activity since the file was deleted.

**CAUTION**

Avoid using ASSIGN, JOIN, and SUBST in conjunction with the delete tracking method.

## Installing deletion protection

If you have plenty of free disk space, use the delete sentry method. One way to install delete sentry is to load UNDELETE using the /S switch. To install delete sentry protection for drive C, for example, add the following statement to your AUTOEXEC.BAT file:

```
undelete /sc
```

If, like me, you tend to have full disks, use the delete tracker method. To install delete tracker protection for drive D, add the following statement to AUTOEXEC.BAT:

```
undelete /td
```

Delete tracker records details on only a limited number of deleted files. The default number of files depends on the size of the hard disk and ranges from 100 on a 20MB drive to 300 on 30MB drives and larger. A 300-entry file consumes less than 60K of disk space. You can explicitly specify the number of entries (in the range from 1 to 999) when you install UNDELETE using the syntax entries. To track the maximum number of files on drive D, for example, revise the entry in AUTOEXEC.BAT as follows:

```
undelete /td-999
```

## Understanding UNDELETE.INI

When you install UNDELETE, settings are recorded in the UNDELETE.INI file in the DOS directory. Rather than remember cumbersome command switches, you can edit UNDELETE.INI to configure your settings and then enter the following command to install UNDELETE using these settings:

```
undelete /load
```

The following file is an example of an UNDELETE.INI file:

```
[configuration]
archive=FALSE
days=7
percentage=20
[sentry.drives]
C=
D=
[mirror.drives]
C=
D=
[sentry.files]
s_files=*.* -*.TMP -*.VM? -*.WOA -*.SWP -*.SPL -*.RMG -*.IMG
-*.THM -*.DOV
[defaults]
d.sentry=TRUE
d.tracker=FALSE
```

The UNDELETE.INI file contains the following five sections:

[configuration]

[sentry.drives]

[sentry.files]

[mirror.drives]

[defaults]

The [configuration] section contains three entries that define whether the archive bit is set when the secret copy is made, the total number of days deleted files are retained in the \SENTRY directory, and the maximum percentage of the total drive space the files in the \SENTRY directory can consume.

The [sentry.drives] section specifies the drives that will be protected by using the delete sentry method when that method is installed. If the delete tracker method is used, this section is ignored. Every drive letter is specified on a separate line and is followed by an equal-sign character. The following three entries, for example, protect drives C, D, and F:

```
C=
D=
F=
```

Use the [sentry.files] section to control which files are monitored and protected when the delete sentry method is used. This section is normally a list of filemasks. Precede a filemask with a minus sign if you want to exclude a set of files. The following entry, for example, protects all files except files with a BAK extension or an extension beginning with TP:

```
*.* -*.BAK -*.TP?
```

The [mirror.drives] section specifies the drives that are protected by using the delete tracker method when that tracker method is installed. (The name is a legacy from the time when Central Point Software, the original developer of UNDELETE, distributed the product.) Use the same syntax as in the [sentry.drives] section.

The [defaults] section controls which method of file protection is implemented when you execute the UNDELETE /LOAD command. One entry must be set to false and the other to true, as shown in this example:

```
d.sentry=FALSE
d.tracker=TRUE
```

**NOTE** UNDELETE, as a minimum, always protects the drives specified in the UNDELETE.INI file, even if fewer drives are specified with the /S or /T switch when UNDELETE is loaded.

If you edit the UNDELETE.INI file, you must unload UNDELETE by using the /U switch and then reload it by using /LOAD. If you loaded another TSR after UNDELETE, you must reboot for the new configuration to take effect.

## DEFRAG — yes, DEFRAG

If you do not use the UNDELETE TSR, your chances of successful data recovery using after-the-fact techniques improve significantly if your disk is not fragmented. Unassisted file recovery is much simpler when a file's data is stored in adjacent (or contiguous) clusters.

Use DEFRAG on a frequent (probably weekly) basis to keep files unfragmented. It not only enhances your chances of data recovery but also improves your system's overall performance. See Chapter 4 for a full discussion of fragmentation.

## Check your disks

Run CHKDSK at least once a week and, preferably, once a day, just to ensure the integrity of the file allocation table. The sooner you know about a file-structure logic error, the better.

Unfortunately, DOS doesn't provide a real disk-analysis tool that verifies the physical integrity of the disk. You may consider supplementing the DOS tools with a third-party disk utility — I use Norton Disk Doctor, which is bundled with Symantec's Norton Desktop for Windows.

## The case of the missing MIRROR

Microsoft surprisingly decided to drop the MIRROR utility introduced in DOS 5. New commands replace much of the MIRROR functionality, but two important MIRROR features are missing from the replacement tools — the capacity to make a copy of the hard disk partition table and record unformat details for each hard drive.

MIRROR is available on the DOS 6 Supplemental disk, and I suggest that you use it, but only for saving the partition and hard drive unformat information.

One of the worst system tragedies occurs when a drive's partition table is accidentally scrambled. The damage can be caused by a program bug that writes to the wrong area of memory or by an unwise user directly editing disk sectors with DEBUG.

NOTE

You can protect your partition table from accidental — and intentional — destruction by using the anti-virus tool VSAFE (discussed later in this chapter).

To save the partition table, execute this MIRROR command:

```
mirror /partn
```

A copy of the hard disk partition table is then stored on a floppy disk, in the file PARTNSAV.FIL.

Every DOS 6 user should also use MIRROR to make copies of the FAT and root directory for each hard drive. This data is vital for recovery from an accidental disk format and is stored in some hidden files on the hard disk.

To save the FAT and root directory details for drives C and D, add the following MIRROR command to AUTOEXEC.BAT:

```
mirror c: d:
```

## Recovering Lost Files and Data

Accidents happen. If you delete a file by accident or format a drive by mistake, or if an errant program destroys your hard disk partition table, you may still be able to recover the data by using one of the DOS data-recovery tools.

### Undeleting files

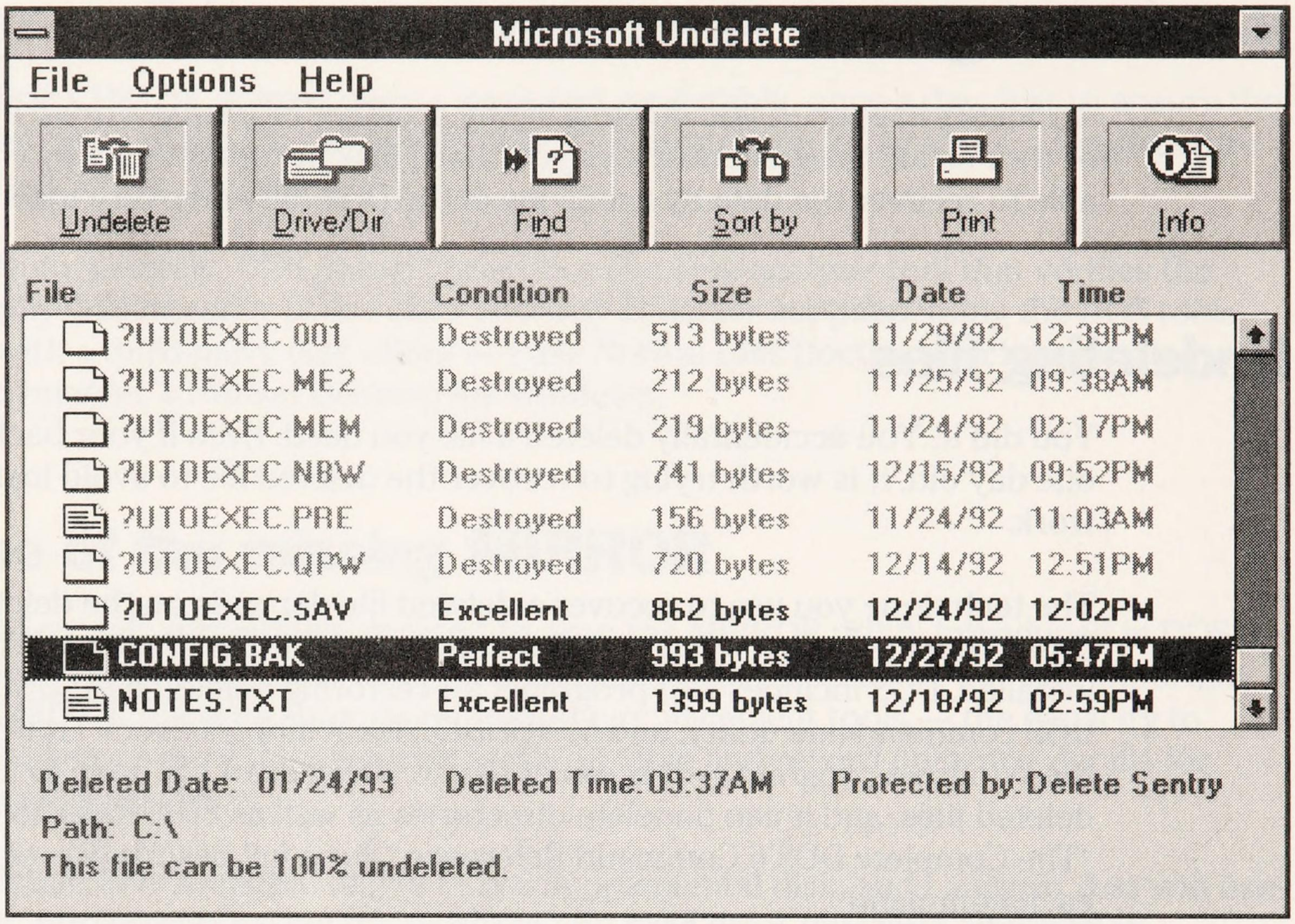
You did it. You accidentally deleted a file you need. Even if your backup is only one day old, it is worth trying to recover the deleted file to avoid losing a day's work.

The technique you use to recover a deleted file depends on the deletion-protection method you have implemented and whether you have Windows installed. DOS includes two programs for restoring deleted files: UNDELETE, a DOS command-line utility, and MWUNDEL, for Windows users. I much prefer the Windows program — it is easier to use, it can search a drive to look for deleted files, and it can undelete directories as well as files. Refer to Part IV, "The Complete DOS 6 Command Reference," for a full description of the use of each command.

Figure 3-8 shows the main MWUNDEL window listing all the deleted files in the active directory. By clicking on the Info button, you can display additional information about the selected file.

Every file in the list includes a Condition field that indicates your probable success in restoring the file. Every file has one of the following conditions:

| | |
|---|---|
| Perfect | You can undelete the file automatically. This category is reserved for files that have been protected with the delete sentry method. |
| Excellent | Because all the file's clusters are consecutive and none of them has been used, the file in all likelihood can be recovered successfully. |
| Good | It seems that all the file's clusters are available, although they are not contiguous. You should be able to undelete the file. |
| Poor | The first file cluster is being used by another file. The chances of recovery are slim to none. |
| Destroyed | All the file's clusters have been reused. You have no chance of recovering the file. |
| Recovered | You have undeleted the file in this session. |



| File | Condition | Size | Date | Time |
|---|---|---|---|---|
| ?UTOEXEC.001 | Destroyed | 513 bytes | 11/29/92 | 12:39PM |
| ?UTOEXEC.ME2 | Destroyed | 212 bytes | 11/25/92 | 09:38AM |
| ?UTOEXEC.MEM | Destroyed | 219 bytes | 11/24/92 | 02:17PM |
| ?UTOEXEC.NBW | Destroyed | 741 bytes | 12/15/92 | 09:52PM |
| ?UTOEXEC.PRE | Destroyed | 156 bytes | 11/24/92 | 11:03AM |
| ?UTOEXEC.QPW | Destroyed | 720 bytes | 12/14/92 | 12:51PM |
| ?UTOEXEC.SAV | Excellent | 862 bytes | 12/24/92 | 12:02PM |
| CONFIG.BAK | Perfect | 993 bytes | 12/27/92 | 05:47PM |
| NOTES.TXT | Excellent | 1399 bytes | 12/18/92 | 02:59PM |

Deleted Date: 01/24/93 Deleted Time: 09:37AM Protected by: Delete Sentry

Path: C:\

This file can be 100% undeleted.

**Figure 3-8**: A list of deleted files displayed by MWUNDEL.

Neither UNDELETE program can undelete files categorized as poor or destroyed.

Note that any file listed with a question mark as the first character was not protected by delete sentry or delete tracker when it was deleted. You can undelete these files only by using the DOS method; these files will never have a classification better than Good.

Unlike MWUNDEL, the DOS command-line utility UNDELETE is not very forthcoming in providing information about deleted files. By combining two switches, however, you can get UNDELETE to list deleted files by category. To see a list of all the deleted files that can be recovered only by using the DOS method, enter the following command:

```
undelete /list /dos
```

Filenames listed with a double-asterisk prefix cannot be recovered. Replace the /DOS switch with /DS or /DT to list the files protected by delete sentry and delete tracker, respectively.

### How DOS Undeletion Works

When a file is deleted (without deletion-protection installed), DOS replaces the first character of the filename with the sigma character ( ). DOS does not list any files that begin with sigma. All the clusters used by the file are marked as available in the file allocation table (FAT). The important point is that no data is deleted.

Because the file's clusters are marked as available, DOS can reuse them to create another file or to expand the size of an existing file and thereby destroy the deleted file's data. The good news is that, because DOS typically uses clusters beyond its most recently saved data cluster, for a while at least, there is a reasonable chance that the deleted file's clusters will remain intact. The longer the file remains deleted, however, the more likely the file's data will be overwritten. Always try to recover an accidentally deleted file as soon as possible after the accident.

DOS maintains the deleted file's name (minus the first character), its starting cluster number, and the number of clusters allocated to the file. By searching a directory for files that begin with the sigma character and analyzing the directory entry, an undelete program can reconstruct the file — assuming that the clusters are still available. The chances of recovering a file are greatly enhanced when the file occupies only a single cluster or when all the file's clusters are stored in a contiguous area. You therefore enhance your chances of recovering larger deleted files by keeping your disk defragmented. If there are fewer contiguous free clusters than the file was using, you cannot recover the file.

If you choose to restore a file, the undelete program prompts for a replacement first character and then modifies the FAT to indicate that the file's cluster (or clusters) is in use.

***Note:*** Some third-party file-recovery utilities enable you to reconstruct a file by browsing through all the unused clusters on a drive and selecting the ones that seem to be part of the file. This technique can be successful with ASCII files, but it is very tedious with binary files.

## Recovering from an accidental disk format

If you accidentally format a disk, you can try to recover it using the UNFORMAT command. Just issue the command that identifies the drive to be unformatted, as in this example:

```
unformat a:
```

There is a chance you can unformat a disk that does not contain the hidden unformatting files. UNFORMAT attempts to reconstruct the missing data. Don't expect miracles, though — the process is far from fool-proof. The DOS UNFORMAT command, in fact, is less reliable than many competitive third-party products, such as PC Tools and the Norton Utilities. All these programs use similar techniques, but — as my father says — "Some are dumber than others." If the data is important and you know that no mirror files exist, you should consider using a third-party unformatting tool rather than the DOS UNFORMAT command.

UNDOCUMENTED FEATURE

You can determine easily whether a disk contains the hidden mirror files by using the /J switch. To check whether the floppy disk in drive A includes the unformat information, for example, enter the following command:

```
unformat a: /j
```

UNFORMAT must try to rebuild the FAT. First, it scours the disk, cluster by cluster, looking for data that resembles a directory file. This process can take as long as 30 minutes for a floppy disk. The directory files are then recovered. Unfortunately, all the root-level directory entries are destroyed by the FORMAT command, so UNFORMAT assigns these root-level directories the names SUBDIR.1, SUBDIR.2, and so on. All lower level subdirectory names can normally be recovered.

GEM

## How Unformatting Works

Most versions of the FORMAT command (including the version in DOS 6) by default do not destructively format the entire drive. They simply rebuild the FAT and the root directory. All other sectors on the disk remain intact.

***Note:*** Some OEM versions of DOS perform destructive formats, including Compaq DOS through v3.2, AT&T DOS through v3.1, and some versions of Burroughs' DOS.

To unformat a disk, the FAT and root directory must be restored. To make life easier, the FORMAT command tries to save all this information in two hidden files: MIRORSAV.FIL and MIRROR.FIL. If you try to view these hidden files on a newly formatted disk, you won't find them. Because the files are created before the FAT and root directory are erased, the file details disappear with all the other files that were on the disk before formatting.

MIRORSAV.FIL, which is always located on the last available cluster of the disk, is small enough to fit on a single cluster. At the beginning of the file is a special string composed of the characters AMSESLIFVASRORIMESAEP, which, when you read it backward, translates loosely to PLEASE MIRORSAV.FIL SESAME. The remainder of the file contains information that identifies the location and size of the MIRROR.FIL file. MIRROR.FIL contains the copy of the FAT and the root directory.

When you execute the UNFORMAT command, the program scours the disk, starting at the last cluster and working backward, looking for the special string. When the special string is located, UNFORMAT reads the location of the MIRROR.FIL file, jumps to the starting cluster, and retrieves the FAT and root directory information. Just like magic. The more a formatted disk is used, of course, the less chance you have of successfully unformatting it. Eventually, the data in the mirror files and in the original files will be overwritten.

If you try to format a disk that is full or that has very little free space, FORMAT cannot create the two mirror files. When this occurs, FORMAT warns you that the UNFORMAT details cannot be saved and prompts for confirmation before proceeding.

If you like to experiment, format an old disk that contains some (no longer needed) files. Then immediately unformat the disk and use UNDELETE to recover the mirror files. You can browse these two files with a binary viewer or DEBUG and see the secrets for yourself.

After recovering the directories, UNFORMAT uses the same techniques as UNDELETE to recover the files in every directory. UNFORMAT cannot recover root directory files or any fragmented files. This instance is just one more that illustrates the advantages of keeping your disks unfragmented.

If you want to use UNFORMAT on a disk that is missing the mirror files, you may consider using the /TEST switch. This switch instructs FORMAT to perform a dry run and not make the modifications, and it gives you a good estimate of the success of the unformat.

Some of UNFORMAT's additional limitations are shown in this list:

- You cannot unformat a disk that has been formatted with the /U switch.
- You can unformat only drives with 512, 1,024, or 2,048 bytes per sector. Note that you can determine a drive's sector size with MSD.
- You cannot unformat network drives.

## Restoring hard disk partition data

UNDOCUMENTED FEATURE

If you used MIRROR to make a copy of the hard disk partition table (discussed earlier in this chapter), you can restore it with the UNFORMAT command by using the /PARTN switch. In most cases, if the partition table is damaged, the system will not boot, so it is a good idea to keep the partition file PARTNSAV.FIL, and UNFORMAT.COM, on a bootable floppy disk.

To restore the partition table, just enter the following command:

```
unformat /partn
```

After the partition table is restored, reboot and — hopefully — your system is back in action.

# Deliberate Data Destruction

To destroy files that contain confidential or personal information, you should do more than just delete the file or format the floppy. As the previous sections of this chapter have shown, there are ways and means to recover "deleted" data.

DISK

If you really want to erase files and make sure that no one can recover them, use RELDEL, from TARDIS DP Consultants, or PDELETE, part of Patriquin's Hard Disk Utilities. Both of these programs are included on Bob's Shareware Collection disks.

To delete files with RELDEL, simple execute RELDEL and specify a filename or filemask. The following command, for example, will destroy all the files with a PRV extension in the current directory:

```
reldel *.prv
```

One of the many useful switches that make PDELETE much better than DOS's DEL command is /WI, for wipe. When you use the /WI switch, PDELETE erases the file and destroys the data the file contained. The following command, for example, deletes the file HATEBOSS.DOC:

```
pdelete hateboss.doc /wi
```

But remember, when it's gone, it's really gone.

## Vital Virus Measures

As with dieting, there is much more talk about viruses than there is action. Unless you live in a cave, you have heard horror stories about computer viruses. Although the truth is that viruses are a reality and they do destroy data, operator error causes much more data loss. Nonetheless, every prudent user should take measures to protect against virus infection, and DOS 6 makes it easy with the following anti-virus tools:

| | |
|---|---|
| MSAV | The Microsoft Anti-Virus utility for finding and removing viruses. This DOS application can be run in full-screen interactive mode or directly from the DOS prompt by using switches. |
| MWAV | Although this Windows version has features that are similar to MSAV, it is much more graphical and provides a couple of useful extra features. |
| VSAFE | This anti-virus TSR monitors disk activity looking for potential virus activity. VSAFE can prevent modifications to some key data areas, such as a disk's boot sector and FAT. |

### The virus phenomenon

In simple terms, a *computer virus* is a program that uses devious techniques to propagate, undetected, from system to system. A program's capability to propagate was the reason that the term "computer virus" was coined. Stimulated by some event, such as executing the program on April 1 or running a program for the tenth time, for example, the virus program makes its presence known. Although the program may do this by simply displaying a message, all too often the virus attempts to destroy your files and data.

There are no good or innocent viruses. All viruses modify your system in some way. In my humble opinion, viruses are written by irresponsible programmers who should be prosecuted. Some 1,200 known virus programs exist. As virus-detection techniques improve, new strains of virus programs are written that strive to avoid detection.

To gain an appreciation for the number of different strains, run the MWAV application in Windows and select Scan⇨Virus List from the menu. MWAV displays a list of all the specific viruses it can detect. Select the Info button to see a full description of the highlighted item (see Figure 3-9).

## Understanding how viruses spread

When a virus-infected program is executed, the virus program is loaded into memory along with the host program. After the virus program is installed in memory, it can — like any program — access drives, files, and printers. Usually, the virus program tries to attach itself to other program files and infect them. These newly infected programs then infect other programs when they are executed, and the virus spreads.

A common misconception is that viruses spread only in EXE and COM files. Viruses are found in all types of executable code, including EXE and COM files, SYS device drivers, Windows DLL files, and overlay files (usually with the letter *O* or *V* in the extension). Viruses cannot spread in nonexecutable files, such as plain ASCII files.

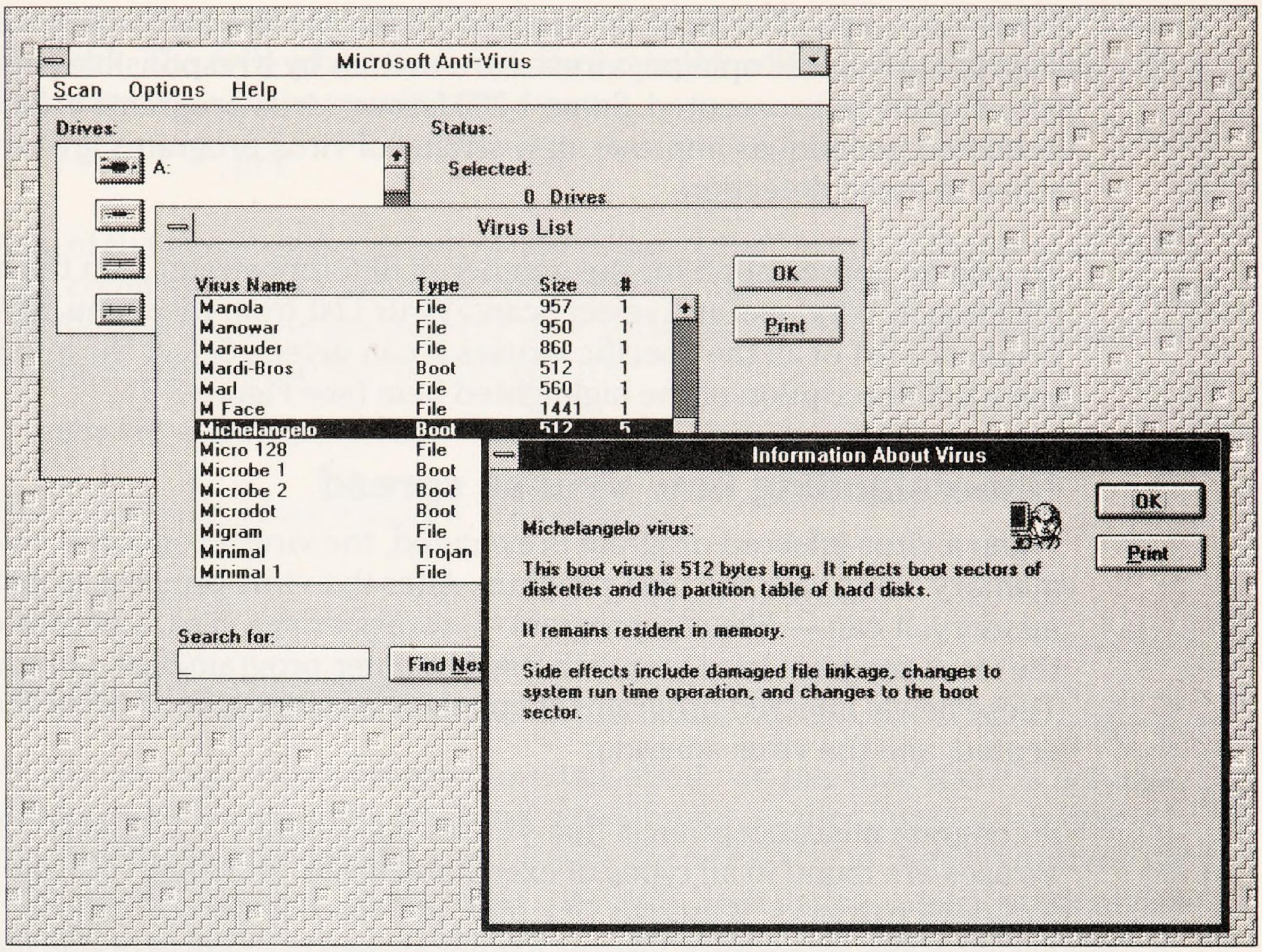
Because a virus spreads only when infected code is executed, you can easily see that your system can become infected only by running programs from other sources. In theory, if you never install new software or run programs from floppies, networks, or via modem, then your system cannot become infected. Unfortunately, most PCs frequently access remote programs (when you load new software from floppy disks, download programs from bulletin boards, and access programs on a network, for example). All these tasks can infect your system.

MWAV identifies the various propagation methods in the Type column (refer to Figure 3-9). Every virus is classified as one of the following types:

| | |
|---|---|
| File | This type adds the virus code to executable programs such as COM, EXE, and DLL files. Some viruses override existing program code rather than append themselves to the end of a program. |
| Boot | A boot virus appends itself to the boot sector of a hard disk or bootable floppy. Every time the system boots from the infected disk, the virus is loaded into memory, where it can propagate to other disks. |

CAUTION

If you are given a boot disk, always check it for viruses before booting your system from it. If you accidentally leave a nonbootable disk in the drive and power up the system, do not press a key and try to boot



| Virus Name | Type | Size | # |
|---|---|---|---|
| Manola | File | 957 | 1 |
| Manowar | File | 950 | 1 |
| Marauder | File | 860 | 1 |
| Mardi-Bros | Boot | 512 | 1 |
| Marl | File | 560 | 1 |
| M Face | File | 1441 | 1 |
| Michelangelo | Boot | 512 | 5 |
| Micro 128 | File | | |
| Microbe 1 | Boot | | |
| Microbe 2 | Boot | | |
| Microdot | Boot | | |
| Migram | File | | |
| Minimal | Trojan | | |
| Minimal 1 | File | | |

Michelangelo virus:

This boot virus is 512 bytes long. It infects boot sectors of diskettes and the partition table of hard disks.

It remains resident in memory.

Side effects include damaged file linkage, changes to system run time operation, and changes to the boot sector.

**Figure 3-9**: MWAV details the characteristics of more than 1,200 known viruses.

again when DOS reports the "non-system disk" error message; immediately remove the disk, power down the system, and try again. Even an aborted boot attempt can infect your system.

Trojan Technically, a Trojan horse program is not a virus. It is a program that disguises its real (destructive) purpose by providing some other function. A seemingly innocent biorhythm program, for example, may include some code to erase your hard disk.

## Potential virus damage

A virus can damage your system in two ways. First, the code used to propagate the virus can modify a program or the boot sector and affect the program's functionality and performance. Even when the virus is removed, the program can malfunction. This type of damage is simply a by-product of the propagation algorithm.

Second, the virus can attempt to damage your system when a trigger event occurs. Most malevolent viruses attempt to damage your system in one of the following ways: destroy the hard disk partition table, destroy or corrupt the FAT, or overwrite files with meaningless characters. Needless to say, all these actions are highly destructive.

## Defending against and identifying viruses

Because a virus program must propagate undetected before it tries to damage your system, a good chance exists that you can identify the infection before any major damage is done.

The following telltale symptoms can provide an indication of virus infection:

- Available memory may decrease.
- Program files may increase in size and reduce free disk space.
- Your system may operate more slowly, become less stable, and crash with greater and greater frequency.

Unfortunately, a memory shortage, a lack of disk space, and system crashes are commonplace; at least on my computers they are. That is why it is important to scan your system frequently to look for viruses.

### Scanning for virus infection

Both MSAV and MWAV can scan your system looking for viruses. Just select the Detect button to scan the selected drive for viruses.

Most known viruses leave a specific pattern of bytes in an infected file, which is sometimes referred to as *virus signatures*. During the scan, the program searches for these signatures in all the files on the selected drive or drives.

You should consider scanning for viruses at least once a week — I do it immediately before running my full backup.

As new virus strains are analyzed, new signatures are identified and added to the signature list. You can update the MSAV and MSAW signature lists via modem by downloading signature updates from a Microsoft-supported bulletin board. Refer to Appendix D of the Microsoft *MS-DOS User's Guide* for further details.

Unlike some third-party anti-virus products, you cannot configure the DOS anti-virus programs to scan an individual directory. The Windows program MWAV, however, supports limited file drag-and-drop in Windows 3.1. To check one or more individual files for viruses, launch File Manager and select the files. Then drag the files off the File Manager window and drop them on the Anti-Virus icon in the Microsoft Tools group. MWAV then launches, scans for viruses in memory, and checks the specific files you dropped on the icon. Neat.

### Where did those CHKLIST.CPS files come from?

Obviously, scanning for signatures identifies only known viruses. The anti-virus programs can optionally use a clever method, *checksums*, for identifying suspicious activity. A checksum is a number value derived by applying a special algorithm to the individual values in a file. When a file is changed, so too is the checksum. By comparing a file's current computed checksum to a checksum created earlier, you can ascertain whether the file has been changed.

To enable checksum scanning, select Options and make sure that the Verify Integrity and Create New Checksums options are checked.

**NOTE** The option settings are stored in an INI file in the DOS directory. The anti-virus programs do not share the same settings: The DOS settings are stored in MSAV.INI, and the Windows settings in MWAV.INI.

Change the options from the main program rather than directly edit the INI files.

The next time you scan a drive, the anti-virus program creates a file named CHKLIST.CPS in every directory. This file records a checksum value for every file in the directory that contains executable code (files with the EXE, COM, or DLL, for example). During future scans, the checksum in the CHKLIST.CPS file is compared with the newly computed checksum.

**NOTE** Some viruses deploy a technique known as Stealth (which is not associated with the QEMM Stealth memory feature) that makes them less vulnerable to detection by using the checksum method. You can instruct the Anti-Virus program to perform a more rigorous scan, however, by selecting the Anti-Stealth item in the Options dialog box.

## Getting rid of those checksum files

The CHKLIST.CPS files provide a valuable anti-virus function, but you can delete them if you want. Rather than laboriously go through every directory and delete every file, you can delete them easily by using one of the following methods:

- If you are a Windows user, just select the option Delete CHKLIST Files from the MWAV Scan menu.
- **DISK** If you don't have Windows, use the PocketD program D.EXE supplied on Bob's Shareware Collection disks, by entering the following command:

```
d chklist.cps /WFzz
```

## Guarding against destructive viruses

You can load the TSR program VSAFE into memory to monitor program and disk activity. VSAFE does not check for specific brands of viruses. It protects the areas usually infected by viruses and provides generic protection against all viruses.

Refer to Part IV, “The Complete DOS 6 Command Reference,” for a full explanation of the switches.

If a program tries to make a suspicious change to the disk (such as modifying the boot sector or changing an EXE file), VSAFE pops up a dialog box to alert you and prompts for confirmation before allowing the change to occur. If you use Windows with VSAFE installed, you must load the DOS 6 program MSAVTSR.EXE. This program enables VSAFE to pop up a Windows-style dialog box to ask for confirmation. Ideally, add the following statement to WIN.INI to load the program automatically every time Windows is started:

```
load=c:\dos\msavtsr.exe
```

When you are using Windows, always disable VSAFE’s option 3, which intercepts all hard disk writes and prompts for approval before proceeding — Windows frequently writes directly to the hard disk, and you will never get any work done because of all the confirmation prompts.

## Removing viruses from an infected system

Both MSAV and MWAV can remove virus infections from the system. This process is referred to as *cleaning*. To remove viruses from your system, simply select the Detect and Clean button.

If the Anti-Virus program detects that a virus is in memory, take the following steps:

1. Turn off your computer.
2. Reboot the computer from a bootable floppy disk that you know is virus free.

   **NOTE** Obviously, you must prepare for this procedure by creating a bootable floppy. If your hard disk includes DoubleSpace drives, this floppy should include the appropriate drivers for DoubleSpace (see Chapter 4). Also, copy the MSAV.EXE, MSAV.INI, and VSAFE.EXE files from your DOS directory for the next step. Execute MSAV to make sure that the floppy disk is virus free and then cover the write-protect tab on the floppy disk to prevent more changes to the disk.

3. Run the anti-virus program MSAV from the floppy and select the Detect and Clean option.
4. Reboot the computer from the hard disk, insert the bootable floppy into a disk drive, and install VSAFE (discussed earlier in this chapter) in memory from the floppy.

5. Some of the files that were infected may be damaged and unusable. If so, you must reload the files from the backup. Be aware that the files on the backup can also be infected. As soon as you have restored the files, run a thorough scan to ensure that the restored files are not infected.
6. Virus-scan all your floppy disks.
7. Immediately warn other computer users with whom you have been in contact, especially people with whom you have exchanged files. If you are a network user, contact your network administrator.

During the next few weeks, you should continue to run VSAFE in a high-protection mode to alert you against reinfection from the same source. You should also perform a complete scan of all your hard disks daily.

## Comparing MSAV and MWAV

Although the two MSAV and MWAV anti-virus programs are similar, a few features are available in one that aren't in the other. This list shows the main differences:

- MSAV can be executed from the command line and supports a variety of switches for automated running. This capability makes MSAV well suited to unattended execution or inclusion in AUTOEXEC.BAT.
- MSAV supports the storing of information in a report file with the /R switch.
- MWAV includes an option to delete all the checksum files from a drive.

## On-line virus information

If you are interested in more information about viruses and you are a member of the CompuServe Information Service, I recommend that you access the excellent Virus Forum by entering the command **GO VIRUSFORUM**. This forum is managed by McAfee Associates, the much respected virus busters. Also, to get the latest updates to MSAV and MWAV, access the Central Point Software forum by entering the command **GO CENTRAL**.

# On a Lighter Note

Data loss is a serious matter we all take too lightly. In keeping with that spirit, I thought you may be interested in some trivia.

If you suffer a system problem and lose valuable data, suddenly everything seems to remind you of your woes. Even movie titles remind you of data loss. Here are but a few:

Indiana Jones and the PC of Doom

Discablanca

Star Wars: The Disk Strikes Back

Crashanova

Here are some CDs (or *records,* for you old-time mainframers) that should constantly remind you to look after your precious files:

Fifty Ways to Lose Your Data, by Paul Simon

Bit out of Hell, by Meat Loaf

Pray, by Hammer

Get Back(up), by the Beatles

Backing Up Is Hard to Do, by Neil Sedaka

---

## *Summary*

This chapter described many of the preventive measures you can take as insurance against data loss and some ways to recover data if disaster strikes. The chapter discussed these subjects:

- How to select files to be backed up by using the point-and-shoot method or by creating an include/exclude list.
- The importance of using the Compare option to verify the integrity of a backup.
- How to restore files by using the backup catalog, including how to reload or rebuild a catalog when it has been removed from the hard disk.
- Essential data-protection precautions, including deletion protection with UNDELETE and saving hard disk partitions with MIRROR.
- Data-recovery procedures using UNDELETE, UNFORMAT, and MIRROR.
- Understanding how viruses spread and how MSAV and MWAV can identify and eradicate a virus infection.

Chapter 4 begins Part II, “Secret Weapons for Your Hardware.” The chapter explores hard disks and explains the all-new DoubleSpace file-compression tools.

# Part II:
# Secret Weapons for Your Hardware

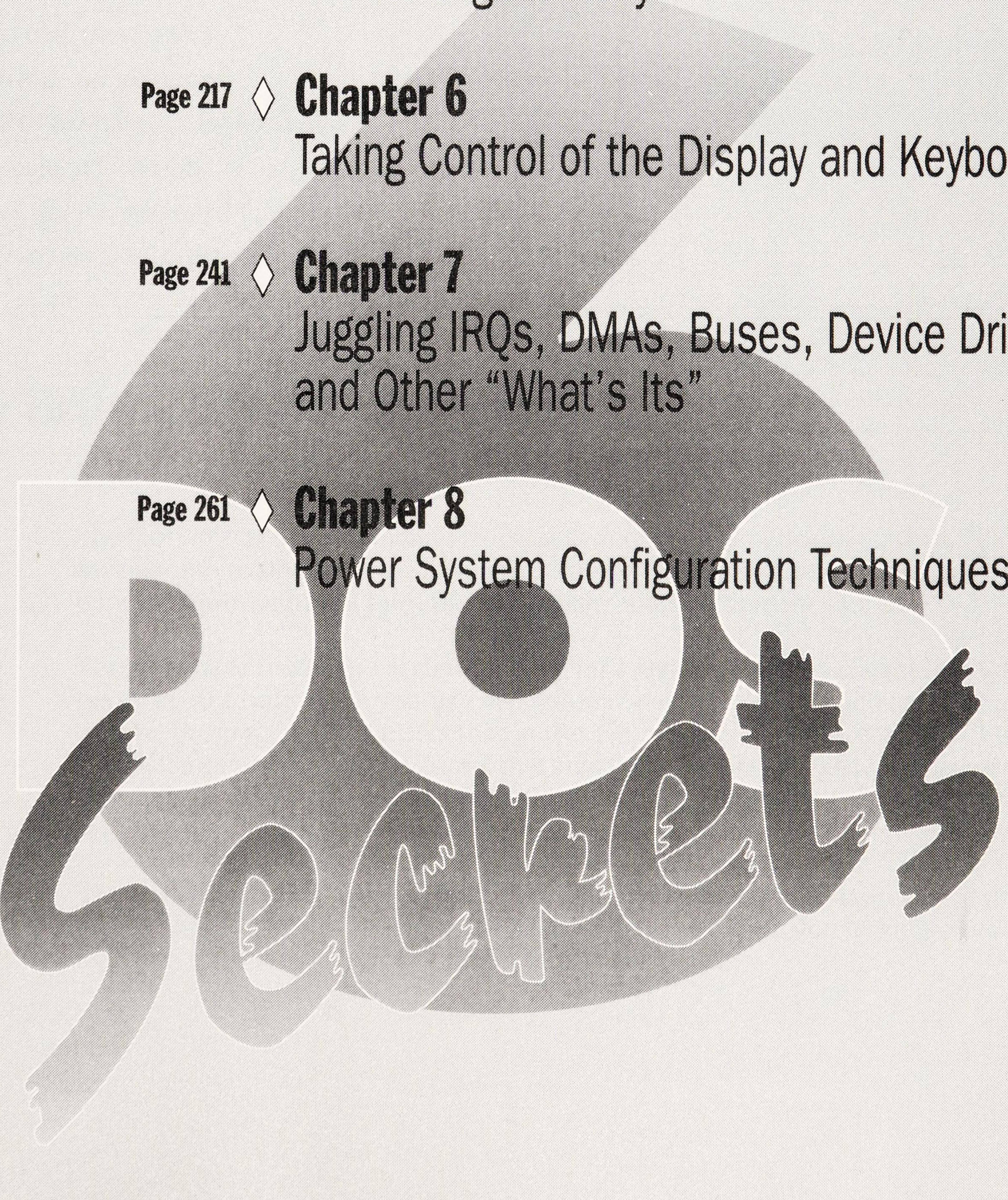
Page 109 ◊ **Chapter 4**
Making Your Hard Disk Faster, Bigger, and Safer

Page 159 ◊ **Chapter 5**
Maximizing Memory

Page 217 ◊ **Chapter 6**
Taking Control of the Display and Keyboard

Page 241 ◊ **Chapter 7**
Juggling IRQs, DMAs, Buses, Device Drivers, and Other "What's Its"

Page 261 ◊ **Chapter 8**
Power System Configuration Techniques



## Chapter 4

# Making Your Hard Disk Faster, Bigger, and Safer

### *In This Chapter*

- Hard disk mechanics
- Data storage schemes
- How to prepare a new hard disk
- The role of CHKDSK
- Ways to increase hard disk speed
- How to use DoubleSpace to make your hard disk bigger

## The Brawn Behind the Brain

If the microprocessor is the PC's brain, the hard disk is its muscle.

Short of buying new hardware, the most effective way of boosting your PC performance is to fine-tune the hard disk. You often can double or triple the effective throughput of hard disks by taking a few simple measures.

You may have heard the argument that hard disks have a significant impact on performance only when you use disk-intensive applications, such as database or CAD programs. The truth is that in the 90s, *all* major software applications are disk-intensive, even programs that traditionally were disk-shy. Word processors access dictionaries, spreadsheets access add-ins and overlay files, and as for Windows programs — well, you know. . . .

The point is that any user today can benefit from a faster and larger hard disk. In this chapter, you first learn how hard disks work. Then you learn how to use DOS 6 to improve the performance of your hard disk.

## The Mechanics of a Hard Disk

One of the few mechanical components in a PC — and clearly the most important — is the hard disk.

It is important that you acquaint yourself with the physical components of the hard disk, including the disk platters and disk heads, and also that you understand the role of the hard disk controller. If you understand the basic mechanics of a hard disk, the whole process of recognizing, diagnosing, and fixing performance problems is simplified. In other words, to be a DOS Jedi, you need to know how a hard disk works.

### Inside a hard disk

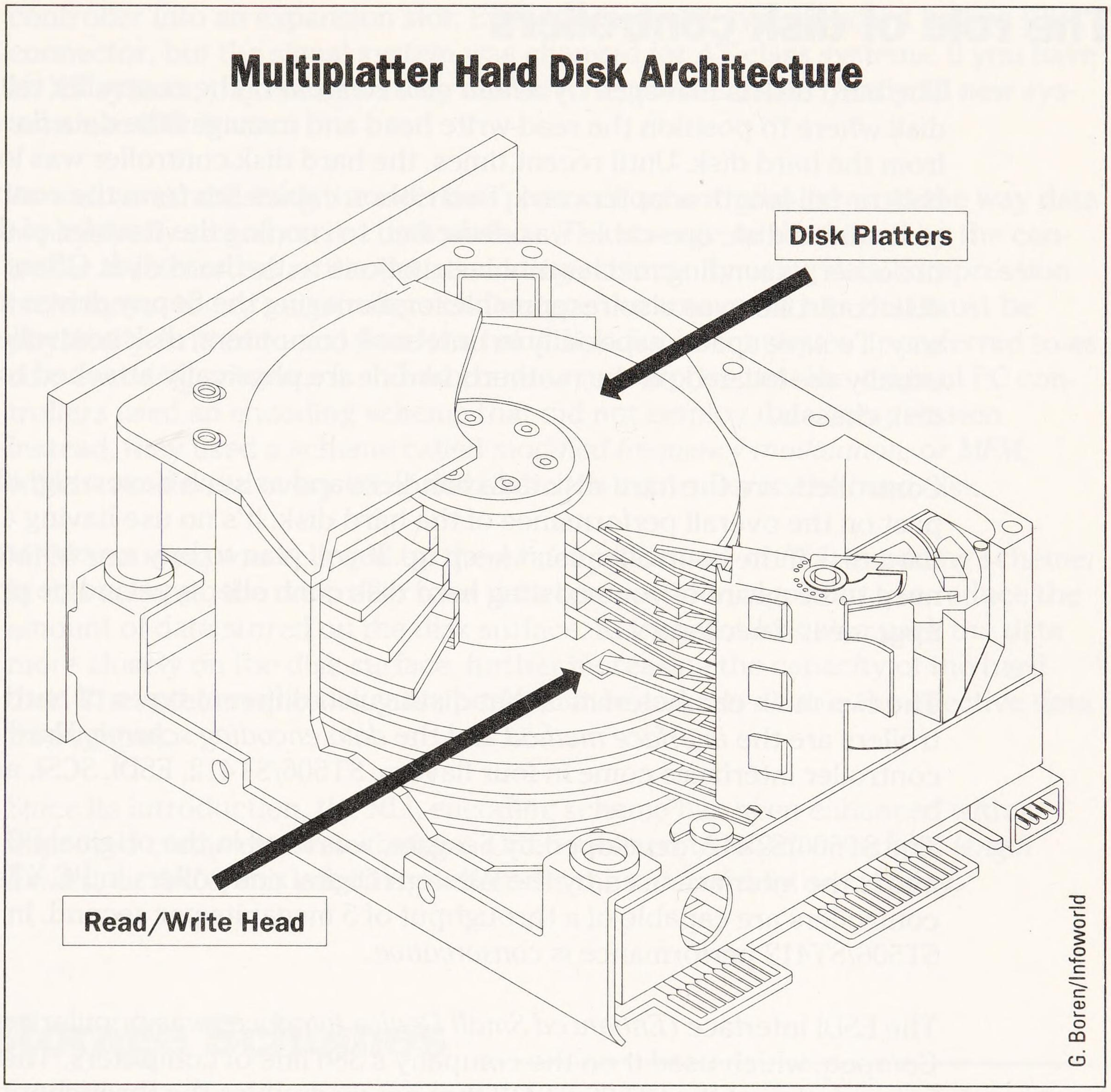
Hard disks come in a variety of shapes and sizes, but they all share some common elements and characteristics. Figure 4-1 shows the inside of a typical hard disk.

The largest component of a hard disk is the *platter.* A platter is a hard circular disk (hence the name *hard disk*), usually made of aluminum. The number of platters in a disk varies but typically ranges from four to sixteen. The platters are mounted in a column, like a stack of records.

A disk platter is similar to a floppy disk. Its surface is coated with a magnetic oxide that is used to store the data. To maximize the storage capacity of the hard disk, both the top and bottom sides of the platter are used. The top side is referred to as *side 0* and the bottom side as *side 1.* You may have noticed that the FORMAT command reports which side, 0 or 1, is being formatted when you format a floppy disk.

Data is transferred to and from the hard disk by a disk *read-write head.* The read-write head is located on a very light armature that pivots across the platter. When placed very close to the platter surface, the head can read or modify the tiny electrical charges in the magnetic oxide — similar to the way a tape recorder reads and records sound on a digital cassette tape.

When data is written to or read from the hard disk, a read-write head must be positioned directly over the appropriate location on the disk. One of the main time delays that occurs when the system accesses data on a hard disk is incurred when the hard disk moves a read-write head to the correct position over the platter surface. To minimize such delays, hard disks have one read-write head for every platter surface. A hard disk with ten platters, for example, has twenty read-write head armature assemblies. This avoids the delay that would occur if an armature had to move from one platter to another. Some of the most



G. Boren/Infoworld

**Figure 4-1:** The inside of a hard disk.

modern disk designs actually have two heads for each surface. Although this offers potential gains on performance, it presents significant design challenges and therefore has not been widely adopted.

A hard disk is not complicated to understand but is challenging to engineer. In a modern hard disk, the disk platters rotate at 3,600 revolutions per minute, and the read-write heads move across the surface of the hard disk just one ten-thousandth of an inch above the platters. If a head were to touch the surface (called a *head crash*) or if a head were not positioned correctly, the data would be at best unreliable and at worst destroyed. Needless to say, any dirt or particles in a hard disk wreak havoc. Hard disks therefore are sealed and have a special filtered air inlet.

## The role of disk controllers

The hard disk is managed by a *hard disk controller.* The controller tells the hard disk where to position the read-write head and manages the data flow to and from the hard disk. Until recent times, the hard disk controller was located on a half- or full-length adapter card. Two ribbon cables ran from the controller card to the hard disk; one cable was dedicated to running the transfer of data and the other to sending mechanical instructions to the hard disk. Often, the hard disk controller was also responsible for managing the floppy drives in the same way. To save space, especially in notebook computers, disk controllers now usually are located on the motherboard or are physically attached to the hard disk chassis.

Controllers are the hard disk data handlers and as such have a significant impact on the overall performance of the hard disk. It's no use having a very fast hard disk if the controller can't keep up. If you plan to buy a new hard disk, you need to be aware of your existing hard disk controller type and be prepared to upgrade it if necessary.

The two main characteristics that distinguish different types of hard disk controllers are the *interface method* and the *data encoding scheme.* Hard disk controller interfaces come in four flavors: ST506/ST412, ESDI, SCSI, and IDE.

The ST506/ST412, developed by Seagate, was used in the original PC hard disk and is the interface used by the Western Digital controllers in PC XTs. Such controllers are capable of a throughput of 5 megabits per second. In a word, the ST506/ST412 performance is *conservative.*

The ESDI interface *(Enhanced Small Device Interface)* was popularized by Compaq, which used it on the company's 386 line of computers. The ESDI really is a variation of the ST506/ST412 that nearly doubles the throughput. Note that an ESDI controller cannot be used on a disk designed for the ST506/ST412.

The emerging interface standard is SCSI *(Small Computer System Interface).* A SCSI (pronounced *scuzzy*) controller can control as many as eight devices, including hard disks, CD-ROMs, and tape drives, and is designed primarily for high-performance computers with a 386 or better microprocessor. The throughput can exceed 20 megabits per second.

The IDE controller interface (*Integrated Drive Electronics*) was originally developed in the mid-80s to save to an expansion slot. ESDI and SCSI drives are actually IDE drives. As the name implies, an IDE drive has the disk controller directly attached to the hard disk — in other words, the controller and the hard disk are a single unit. The IDE controller attaches to a special IDE connector located on the system board, thereby avoiding the need to plug a separate disk controller into an expansion slot. Even the lowly XT system board had an IDE connector, but the signal system was changed for AT-class systems. If you have an XT system, then you can only use an XT-class IDE drive. Nearly all new systems use IDE drives.

As well as establishing the interface protocol, a controller governs the way data is physically stored on the disk surface. The storage method used by the controller is referred to as the data encoding scheme. By using data compression techniques, today's controllers can reduce the amount of bits that must be physically written to and from the hard disk. Such techniques are referred to as *data encoding schemes.* It should come as no surprise that the original PC controllers used an encoding scheme that did not employ data compression. Instead, they used a scheme called *modified frequency modulation,* or *MFM,* which transferred every bit of data from the controller to the hard disk.

MFM encoding was eclipsed by the *run length limited,* or *RLL,* encoding scheme. In this scheme, the controller uses data compression algorithms to reduce the amount of data stored on the disk surface. RLL controllers also pack the data more closely on the disk surface, further increasing the capacity of the hard disk. These techniques increase performance and can double the effective data throughput.

Since its introduction, the RLL encoding scheme has been enhanced with an improved compression technique. Often referred to as *advanced run length limited,* or *ARLL,* this encoding scheme is the most widely used today.

# Data Storage Schemes

Data integrity and performance are the two primary concerns for a hard disk. The way data is physically stored on the hard disk has a profound effect on both of these issues. When you save a file from your word processor (and by the way, you should save your file more frequently!), the operating system goes through a whole series of computations to decide where to store the data so that it can be retrieved again later. A lot happens when you select File and then Save.

A hard disk is like a warehouse for storing computer data. If items in a warehouse were stored in a haphazard way, havoc would ensue — you'd never be able to find anything. To prevent this confusion, warehouses use numbering or addressing schemes that enable users to locate any item in the warehouse quickly. A warehouse typically is divided up into bays, bays are subdivided into racks, and racks are further subdivided into storage bins or shelves.

Hard disks use a similar scheme, but the nomenclature is different. Disk storage areas are organized by *tracks, sectors,* and *cylinders.*

## Tracks

Because each disk platter continually rotates (usually at 3,600 revolutions per minute), a disk read-write head can access the entire circumference of a thin band of the platter without moving. By moving inward toward the center of the disk or outward toward the edge of the disk, the read-write head can access the entire surface of the platter, reading or writing data in a series of concentric circles. These concentric circles, which are the width of the read-write head, are referred to as *tracks.* A platter can have a thousand tracks or more per inch. Each track is given a unique number, with the outermost track assigned the number 0.

Every platter surface has the same track-numbering scheme. Track 3 on one surface is vertically aligned with track 3 on each of the other surfaces, for example. The term *cylinder* refers to the vertical column of tracks.

## Sectors

On some systems, a single track is large enough to store 27K or more of data. It would be wasteful to consume (or allocate) an entire track for a file smaller than 27K. Storing only one file per track would also pose a problem for files larger than 27K. If a file were 57K in size, for example, 24K of space would be wasted, because a total of three tracks would be needed to store the file.

To make more efficient use of the available space, each track is subdivided into a series of individual sections called *sectors.* Each sector can store 512 bytes of data. The number of sectors in a track depends on the capacity of the track but usually ranges from 17 to 55. Figure 4-2 illustrates a disk platter organized into tracks and sectors.

### Breaking the 1,024 Cylinder Barrier

DOS 6 and all of its predecessors do not support a drive with more than 1,024 cylinders. Most hardware vendors that provide disks using more than 1,024 cylinders also provide a device driver and disk-formatting software that tricks DOS into accessing the additional cylinders.

One strategy is to hook into interrupt 13 and advise DOS that the disk has more heads and fewer cylinders so that the total number of tracks is accurate but the number of reported cylinders is within DOS's range. Many of the newer IDE drives take advantage of the close integration of the controller and the drive to mask the real cylinder count, without having to intercept DOS interrupts.

You may find that the Window's Fastdisk utility (discussed later) is not reliable on systems where the real cylinder count exceeds 1,024.

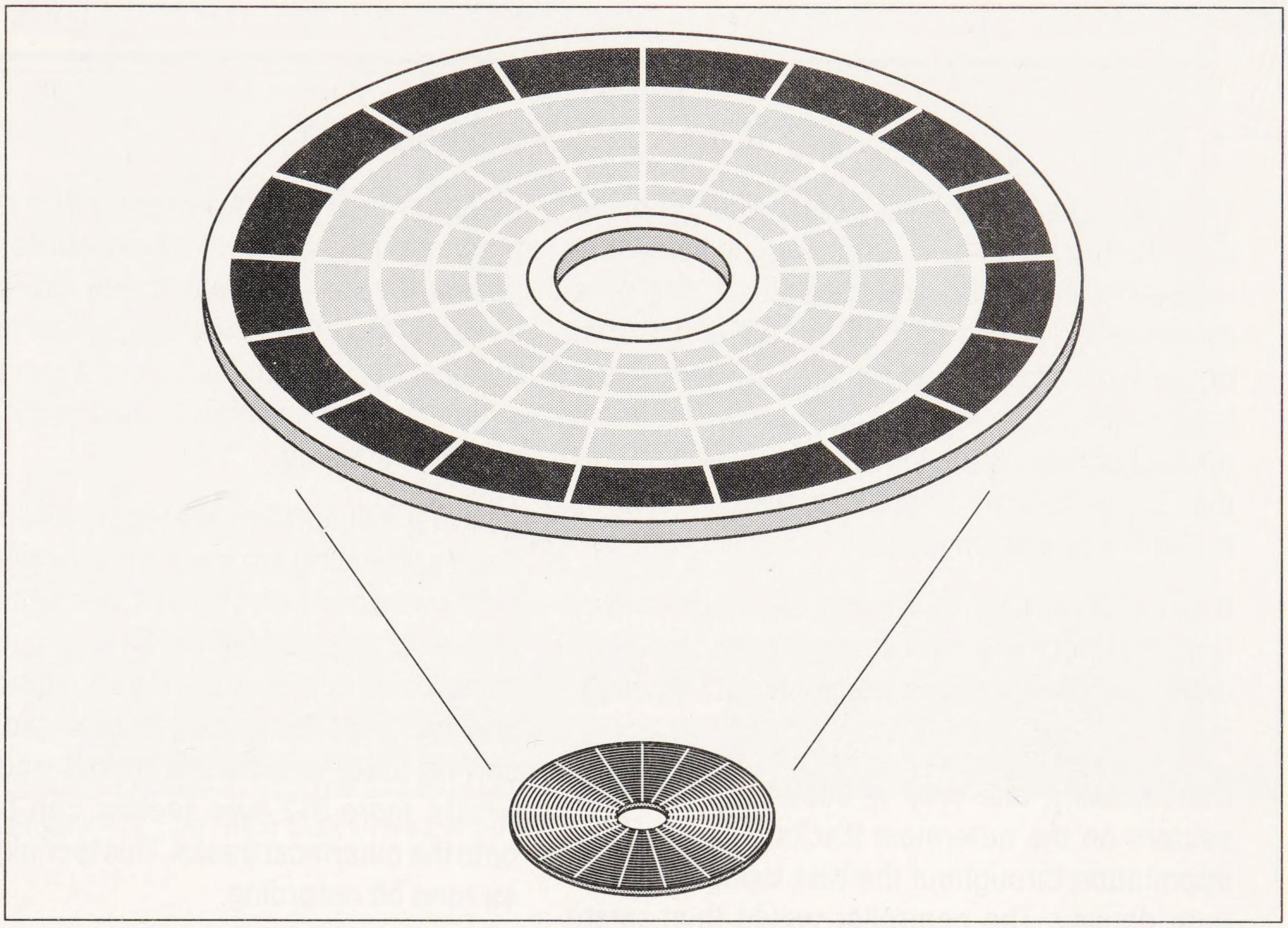


**Figure 4-2:** Disk tracks and sectors.

Although the outermost tracks have a larger diameter and therefore a larger surface area, most disks allocate the same number of sectors to the outermost (longer) tracks as are allocated for the innermost tracks. However, to increase the storage capacity, some newer disks use a technique called *zone bit recording,* which squeezes more sectors onto the outer tracks.

## Clusters

One of the primary functions of DOS is to know which sectors are in use and which are available. Managing all these sectors is a burden. A 300MB hard disk, for instance, has more than half a million sectors.

To make the hard disk more manageable, DOS manages blocks of contiguous sectors together. Each block of sectors is referred to as a *cluster.* The number of sectors in a cluster varies and typically ranges from 4 to 16. In a system configured with 8 sectors per cluster, DOS has to worry about only one-eighth of the storage areas, so performance is significantly improved.

## Zone Bit Recording

The farther away a track is from the center of the disk, the faster the disk surface spins below the read-write head. On standard disks, the data density of the outermost tracks is lower than that of the inner tracks. The sectors are longer but have a lower density, so they contain the same amount of data as the inner sectors. This means that the read-write heads are always handling the same data throughput.

It seems wasteful to reduce the data density (and therefore the total storage capacity of the drive) just to maintain a constant data throughput.

Conceptually, one way to accommodate more sectors on the outermost tracks is to store the information throughout the disk using the same data density. The controller would then rotate the disk at lower speeds when reading or writing the outermost tracks, ensuring that the disk surface moves below the read-write head at a constant rate. Variable-speed disks, however, have slow access times; the system must wait for the disk to speed up and slow down as the head moves from one region to another. It's just not a practical solution.

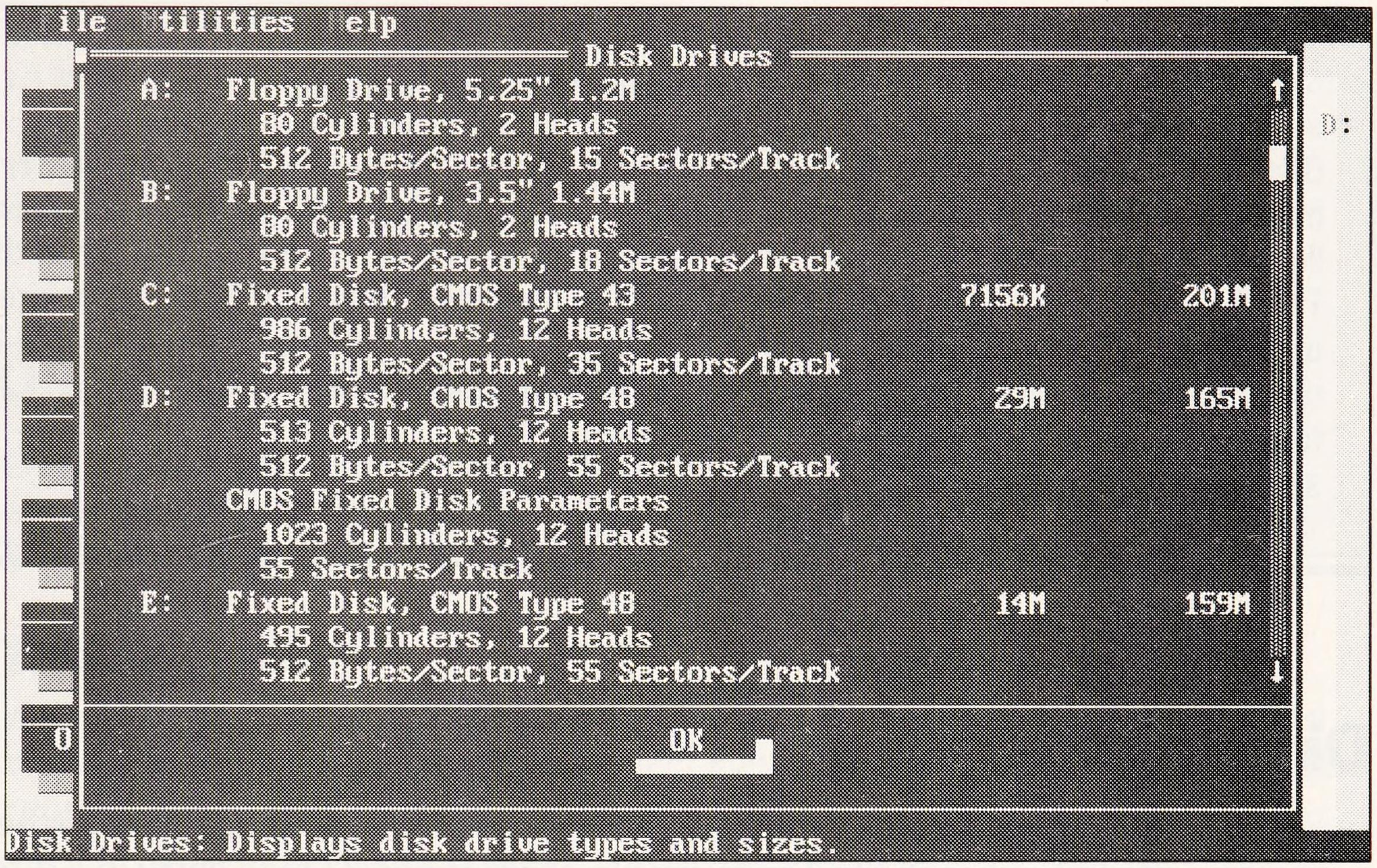
The real solution lies in electronics, not mechanics. By changing the frequency of operation of the read-write head electronics, the system can read and write data at different throughput rates. If the same data density is used throughout the platter and the head frequency is changed, more data can be read from the outermost tracks. In other words, more 512 byte sectors can be squeezed onto the outermost tracks. This technique is known as zone bit recording.

The tradeoff is that space is allocated in single, cluster-size chunks. The larger the cluster size, the more space is potentially wasted. On a system with 8 sectors per cluster, space is allocated in 4K chunks. A 50-byte batch file uses up 4K of space, and a 37K file uses 40K.

DOS can only manage a limited number of clusters (65,518 to be precise), and so the cluster size is larger on high-capacity drives to ensure that all the available disk space can be addressed. If the cluster size is too large, space is wasted. On the other hand, if the cluster size is too small, files become too fragmented and disk throughput is reduced. As a general rule, you should use one of the following cluster sizes: 2,048, 4,096, or 8,192 bytes.

## How to check your drive characteristics

The DOS 6 diagnostic program MSD provides some useful information about your hard disk(s). Execute MSD and select the **D**isk Drives option. When you select this option, the Disk Drives window, shown in Figure 4-3, appears. The characteristics of each disk (including floppy drives) are displayed. DOS reports the number of cylinders (tracks), heads (platter surfaces), bytes per sector, and sectors per track. The total capacity of each logical drive along with the amount of free space is displayed as well.



```
                          Disk Drives
A:   Floppy Drive, 5.25" 1.2M
       80 Cylinders, 2 Heads
       512 Bytes/Sector, 15 Sectors/Track
B:   Floppy Drive, 3.5" 1.44M
       80 Cylinders, 2 Heads
       512 Bytes/Sector, 18 Sectors/Track
C:   Fixed Disk, CMOS Type 43                  7156K       201M
       986 Cylinders, 12 Heads
       512 Bytes/Sector, 35 Sectors/Track
D:   Fixed Disk, CMOS Type 48                    29M       165M
       513 Cylinders, 12 Heads
       512 Bytes/Sector, 55 Sectors/Track
     CMOS Fixed Disk Parameters
       1023 Cylinders, 12 Heads
       55 Sectors/Track
E:   Fixed Disk, CMOS Type 48                    14M       159M
       495 Cylinders, 12 Heads
       512 Bytes/Sector, 55 Sectors/Track
```

```
Disk Drives: Displays disk drive types and sizes.
```

**Figure 4-3:** The MSD Disk Drives window.

## The file allocation table

Every DOS drive has a *file allocation table,* or *FAT.* In fact, because the FAT is so important, DOS maintains two copies of it, just in case one gets damaged. The FAT is responsible for keeping records about every cluster on the drive. At the beginning (or outside edge) of each drive is a boot record, the FAT, and root directory, followed by the general data storage area.

The FAT assigns a unique number to each cluster. The first cluster in the data area (immediately following the root directory) is assigned the value 2. That's right — the first data cluster is numbered 2. The next cluster is assigned number 3, and so on. The FAT can keep track of 65,518 individual clusters.

In the FAT, there is a two-byte *status field* for every cluster. The cluster status indicates that the cluster is in one of the following five states:

- The cluster is unused and available (value 0).
- The cluster is reserved (values FFF0 to FFF6).
- The cluster is bad and should not be used (value FFF7).
- The cluster is the last one in a file (value FFF8 to FFFF).
- The cluster is in use (the value is the cluster number of the next cluster in the chain).

### 12-bit versus 16-bit FATs

Originally, DOS created a FAT that used only 1½-byte fields (12 bits) to store the cluster details, limiting the maximum number of clusters to 4,078. These 12-bit FATS severely limited the capability of DOS to handle larger drives. The only real way to increase the addressable area of a disk was to increase the cluster size, which led to wasted space.

Since the introduction of version 3, DOS has supported both 12-bit and 16-bit FATs. DOS decides which size FAT to use when the drive is formatted. Drives 16MB or less in size use a 12-bit FAT, and all larger drives use a 16-bit FAT.

## Directories

In conjunction with the FAT, DOS uses directories to maintain further information about each file. For every file, there is one 32-byte entry in a directory. Each file's directory entry contains the information listed in Table 4-1.

When a drive is formatted, DOS creates a root directory with a fixed number of entries. The total number of entries depends on the size of the disk but usually ranges from 112 on 360K floppies to 512 on larger hard disks. Obviously, it would be very limiting to be able to store only 512 files on any one disk. The solution lies in directories and subdirectories.

**Table 4-1 Directory File Information**

| *Entry Description* | *Size in Bytes* |
|---|---|
| Filename | 8 |
| Extension | 3 |
| Attribute | 1 |
| Reserved | 10 |
| Time | 2 |
| Date | 2 |
| Starting FAT Entry | 2 |
| File Size | 4 |

When you create a directory, DOS creates a special file in the parent directory to keep track of all the files in that directory. The contents of the special file are organized in the same way as the root directory; however, the number of directory entries is unlimited. There is one directory file for every unique subdirectory on the system.

## How DOS reads, writes, and deletes files

By using the information in the directories and the FAT, DOS can read, write, and delete files reliably. The following sections provide an overview of how DOS performs each of these tasks.

### Reading a file

A lot of activity takes place behind the scenes when you ask DOS to open and read a file. For example, when you select **F**ile and then **O**pen in EDIT, a file open request is passed to DOS.

After DOS receives the request, its first task is to locate the file's directory entry. Suppose that the filename is C:\BUDGET\LOWCASE\USA\NOTES.TXT. DOS goes to the root directory of drive C and locates the subdirectory file named BUDGET. DOS opens the file BUDGET and searches for the location of the LOWCASE subdirectory file. DOS then opens the LOWCASE file and searches for the USA entry. Finally, DOS opens the USA subdirectory file and reads the entry for NOTES.TXT. The directory entry for NOTES.TXT contains the starting cluster number of the file's data. Stage one is then complete.

DOS next switches its attention to the FAT. Using the starting cluster number as an index into the FAT, DOS reads the FAT entry for NOTES.TXT. If the 2-byte status code has a value in the range FFF8 to FFFF (the last-cluster codes), the entire file's data is held in one cluster; DOS simply reads the data from the single cluster, and the file is loaded. However, if the status code points to a second cluster, DOS must read the first cluster and then move to the second cluster's FAT entry. It reads the data in the second cluster, and if the second cluster's FAT entry points to a third cluster, the process continues. DOS reads the data from each cluster in the chain until it reads the file's last cluster — that is, the cluster that has a 2-byte status code in the range FFF8 to FFFF in the FAT. *Phew.*

### Deleting a file

When you delete a file, DOS changes the first character in the directory entry to the sigma character, which has the ASCII value 229. The FAT entries for the file's clusters are then updated with the cluster-available code. That's all there is to it.

Apart from the first character of the filename, no data is actually destroyed when you delete a file. For a thorough discussion of file deletion and undeletion, refer to Chapter 3.

### Writing a file

When you save a new file to disk, DOS tries to use clusters on the innermost unused part of the disk. This strategy increases the chances that the entire file can be saved in adjacent clusters — in other words, not fragmented — and enhances the chances of data recovery using UNDELETE because a deleted file's clusters are not immediately overwritten. If no free space exists in this region, DOS starts to save the file using the first free cluster indicated in the FAT. If the entire file fits in one cluster, DOS marks the FAT entry with a last-cluster code, and the process is complete. If the file is too large to fit in one cluster, DOS locates the next free cluster by scanning the FAT and then records the second cluster entry in the first cluster's status code, thereby creating a *cluster chain.* If the remainder of the file fits in the second cluster, the cluster's FAT entry is marked with the last-cluster code. Otherwise, the process continues until all the data is written to disk. DOS then creates the directory entry for the file and records the starting cluster number in it.

If the file has previously been saved to disk, DOS uses a different methodology. It uses the directory entry and FAT to locate the file's clusters, similar to the way it does when reading a file. DOS attempts to use the same clusters to store the updated data. If the revised file is smaller than the original file, DOS marks the unused clusters as free in the FAT and marks the file's last cluster entry with a last-cluster code. If the revised file is larger than the original, DOS scans the FAT for the next free cluster and extends the cluster chain until the file is completely saved.

## How to Prepare a New Hard Disk

Three levels of preparation are required before a hard disk is ready to store files. These stages are typically referred to as *low-level formatting, partitioning,* and *high-level formatting.*

## Low-level formatting

Every sector on the disk has a unique address. This address is simply a combination of the platter (or head) number, the track (or cylinder) number, and the sector number within the track. At the beginning of every sector, there is a special header section in which, among other things, the sector address is stored. When data needs to be retrieved from the hard disk, the read-write head is positioned over the appropriate track and the sector addresses are read. As soon as the correct address is found, the data is read from the sector that follows. This addressing scheme is referred to as *soft sectoring.*

The low-level formatting process divides the disk into sectors and then records the sector address in each sector header. Low-level formatting also is responsible for verifying the integrity of the disk surface. If any sectors are found to be defective, they are marked as unusable.

## Disk partitioning

Because low-level disk formatting usually is performed at the factory, the first stage of disk preparation in which users typically become involved is disk partitioning. A single physical disk can be subdivided, or *partitioned,* into smaller logical drives. For example, a single 500MB hard disk might be partitioned into three logical drives: drive C with 200MB, drive D with 175MB, and drive E with 125MB. Each of these drives is referred to as a partition.

It is now commonplace for a computer to be delivered ready to run. In some cases, popular software packages, such as Windows, are already installed as well. Rest assured that if your computer arrives with software installed, the disk is low-level formatted, partitioned, and high-level formatted.

Even if you want to use a hard disk as one logical drive, the disk still must be partitioned, albeit with a single partition.

In the past, one of the most common reasons to partition a disk into several logical drives was related to a DOS shortcoming known by many as the "32MB limit." Before the release of DOS 4.0 (or 3.3 for some DOS strains), DOS could only access drives up to 32 megabytes in size. This meant that someone who owned a 100-megabyte disk, for example, could only access the first 32 megabytes. The solution was to partition the disk into multiple drives, with each one being 32MB or less in size. The new size limit for drives is 2 gigabytes (2,048MB). Even with the rapid pace of change in this industry, I suspect that this limit will suffice for some time to come.

Prior to the release of DOS 4.0, a 16-bit number was in the boot record to store sector details, which meant that DOS could only handle 65,536 sectors. Because each sector is 512 bytes in size, the maximum storage space that could be managed was 65,536 x 512 bytes, which is 32MB. With the introduction of DOS 4.0, two 16-bit numbers were used to store sector details in the boot record, shattering the 32MB limit.

Following are the main reasons that you might consider organizing a single physical disk into several partitions:

- Although the vast majority of PCs use the DOS operating system, not all do. Some of the most popular alternatives are UNIX, XENIX, and OS/2. Partitioning enables you to install more than one operating system on a single disk. Each disk partition can accommodate a different operating system.
- With today's large hard disks, subdividing a disk into smaller drives is often done just to make it more manageable or to reduce the space wasted by saving small files to large clusters. Also, really big drives choke DEFRAG (see "Solving memory problems" later in this chapter). Anyone who has forgotten the name of a file and spent hours trying to find it on a 500MB hard disk knows that there are a lot of places to look. Partitioning provides a convenient way to avoid this problem. For example, you might use drive C for all programs associated with word processing and desktop publishing and use drive D for all the other program files, such as spreadsheets and databases.

DOS provides the program FDISK for partitioning a hard disk. FDISK is responsible for storing the disk partition information on the hard disk. This information is stored in the *master partition table,* located in the first sector of the first track on the first platter. Refer to Part IV, "The Complete DOS 6 Command Reference," for more information about using FDISK.

TECHNICAL STUFF

## Huge Disks and Small FATs

Aside from making drives easier to manage, there is one more compelling reason to organize huge disks into several logical drives: to save space. That's right — to *save* space.

The FAT has room to manage 65,518 individual clusters. The number of bytes per cluster is always a value of 2 to a power, as well as an even multiple of 512 bytes (the standard sector size). This results in cluster sizes of 512, 1,024, 2,048, 4,096, 8,192, 16,284, and 32,768 bytes.

Consider the cluster organization of a 1 gigabyte drive — in other words, a 1,073,741,824 byte drive. Because the total number of cluster entries is 65,518, each cluster must be at least 16,389 bytes large. Rounding up to the next available cluster size results in a cluster size of a whopping 32,768 bytes. Imagine every batch file consuming 32K!

Because each drive has its own FAT, dividing a huge disk into several drives can reduce the cluster size and thereby increase the real storage capacity. In this example, if the 1 gigabyte disk is organized into four 250MB drives, the cluster size falls to a respectable 8,192 bytes. Better still, five 200MB drives can use a cluster size of 4,096.

The moral of this analysis is that smaller is sometimes bigger.

The master partition table informs the system of the way the disk is partitioned. It also contains the starting and ending sector of each partition, a code indicating the operating system for each partition, and information about which partition should be used to boot or start the computer. When the computer is first booted, the partition table is inspected and then attention is turned to the boot partition, from which the appropriate operating system is loaded. The partition used to boot the computer is referred to as the *active partition.*

**DOS 6**

You can easily check the status of partition table by executing the following command:

```
fdisk /status
```

This new switch displays the partition settings, detailing the number of installed disks and the logical drives associated with each disk.

Clearly, the partition table is a very important element of the hard disk. If the partition table gets damaged, you may not even be able to boot the computer from the hard disk. Refer to Chapter 3 for information about using MIRROR to keep a copy of the partition table.

**NOTE**

Some disks are configured with an odd number of platters — seven, for example. In such situations, the odd platter, known as the *servoplatter,* is used, but not to store user data. The platter contains sector addressing information, which ensures that any sector can be accurately located even when the platters expand and contract due to operating temperature fluctuations. This accurate location of each sector reduces reseeking and thereby speeds up disk access. Newer *embedded servo* technologies have eclipsed this once popular technique.

## The Role of CHKDSK

As the name suggests, the CHKDSK command (short for *check disk*) checks the integrity of the disk. CHKDSK doesn't check the physical condition of the disk; it merely analyzes the directory entries and the FAT and then identifies inconsistences and problems. If you execute CHKDSK with the /F switch, DOS tries to correct any problems it encounters. Otherwise, it reports problems but makes no attempt to correct them.

CHKDSK is not very adept at recovering data when a problem occurs. Its main focus is to remove the cause of the errors, not to recover data that was lost as a result of those errors. You should consider purchasing a third-party data recovery package, such as Norton Utilities, from Symantec, or PC Tools, from Central Point Software, to name just two. These packages might be worth the purchase price the first time a valuable file is damaged. Nonetheless, CHKDSK does act as a valuable file-integrity barometer and can solve some simple problems.

ERROR MESSAGES DECODED

The following paragraphs explain the main error messages reported by CHKDSK:

- *filename* `is cross-linked on allocation unit` *x*

  Two or more files both claim ownership of a specific cluster. CHKDSK is unable to solve this problem even when it is used with the /F switch. You might be able to solve the problem by copying the files to another drive, deleting the original files, and then copying the files back to their original directories. In all likelihood, one file will be correct and the other will be corrupted. Check both files by loading them into an appropriate application and determine which file is damaged. You probably will need to restore a copy of the damaged file from backup.

  NOTE

  If you encounter cross-linked errors on valuable files, you should consider using a third-party data recovery program such as The Norton Utilities or PC Tools.

- *x* `lost allocation units in` *y* `chains`

  Some clusters are marked in the FAT as being in use, but no file claims ownership of them. That is, they are not in the chain of any file. When you use the /F option, DOS can optionally create one file in the root directory for each lost chain, giving you an opportunity to see what data is stored in these clusters. The files are given standard names, using the syntax `file`*nnnn*`.chk`, where *nnnn* is a sequential number — for example, FILE0001.CHK, FILE0002.CHK, etc.

- `Insufficient room in the root directory`

  The root directory of a hard drive is limited to 512 files. The limit is even lower on floppies and RAM disks. CHKDSK displays this error message when trying to create the FILE*nnnn*.CHK files (discussed previously) from the lost chains. To solve the problem, move some files into a subdirectory and then execute CHKDSK /F a second time.

- `File allocation table bad, drive` *x*`:`

  This message means what it says. The FAT is corrupted and cannot be read. If you regularly use MIRROR to keep a copy of the FAT, as recommended in Chapter 3, you can restore the FAT using UNFORMAT. Otherwise, you should consider using a third-party FAT rebuilding tool. If all else fails, try to copy all of your important files to another disk and then restore your entire drive from backup.

- `First allocation unit is invalid, entry truncated`

  The first cluster specified in a file's directory entry does not exist. When used with /F, CHKDSK changes the file's size to 0, indicating that there is no corresponding data.

- `Has invalid allocation unit, file truncated`

  One of the cluster entries in the file's cluster chain points to a cluster that does not exist. When used with /F, CHKDSK reduces the file's size and marks the last sensible cluster in the chain as the last cluster.

- `Allocation error, size adjusted`

  By comparing the file size in the directory entry with the total number of clusters allocated to the file's cluster chain, CHKDSK determined that too many clusters were allocated. CHKDSK assumes that the file-size entry in the directory is correct. When the /F switch is used, CHKDSK computes how many clusters are needed, changes the FAT status code for the real last cluster and assigns it the last-cluster code, and marks all subsequent clusters as available.

- `Disk error reading FAT` *n*

  `Disk error writing FAT` *n*

  Two copies of the FAT are maintained at the beginning of the drive. This message indicates that one of the copies cannot be accessed. If the other copy were to become damaged, you would have a serious problem. Treat this message as a critical warning. One sure solution is to back up your entire drive, reformat, and then restore the files from backup.

- `CHDIR .. failed, trying alternate method`

  CHKDSK is unable to use the ".." file to traverse to the parent directory. To resolve the problem, CHKDSK goes to the root and searches downward for a directory that contains a directory file pointing to the subdirectory with the missing ".." file.

- `Directory is totally empty, no . or ..`

  A directory does not even contain the standard "." and ".." files. Remove the directory using RD and try CHKDSK again.

- `Cannot recover .. entry`

  There is a problem with the special ".." file. CHKDSK can normally correct these problems when you use the /F switch.

- `Cannot CHDIR to` *directory*`, tree past this point not processed`

  No matter how it tries, CHKDSK cannot access a directory. This is an indication of a corrupted directory file and is best resolved using a third-party data recovery tool.

- `Cannot CHDIR to root. Processing cannot continue`

  A serious corruption of the directory structure has occurred. Again, the problem is best resolved by using a third-party data recovery tool.

- `Unrecoverable error in directory. Convert directory to file (Y/N)?`

  CHKDSK identified a major error in the drive's directory structure. If you press Y, CHKDSK effectively deletes all the files stored in the bad directory. I recommend that you respond by pressing N and try to resolve the problem with a third-party data recovery program.

You may be wondering how problems such as these occur. They can happen when the system loses power in the middle of writing a file — for example, if you turn the computer off by mistake. Other causes include program bugs and computer viruses.

When you run CHKDSK on a DoubleSpace drive, it performs exactly the same checks as it does on a standard drive. After all, the FAT structure and directory entries are the same for both drive types. However, when CHKDSK finishes checking a DoubleSpace drive, DOS automatically invokes the command DBLSPACE /CHKDSK. DBLSPACE /CHKDSK verifies the integrity of the additional DoubleSpace data structures, including the MDFAT, which maps a FAT cluster to the location in the DoubleSpace drive where the compressed cluster is stored.

## Faster, Faster, Faster

You can use many software tools to enhance the performance of your hard disk. The most effective measure is to install a software disk cache, but other alternatives include setting the optimum interleave, using Fastdisk in Windows, and defragmenting files. If you haven't personally taken these measures, the chances are that with a few simple steps, you can increase the speed of your hard disk. Don't just assume that your hard disk is working at full throttle.

### Optimizing the interleave

During low-level formatting, a disk is organized into sectors. You might expect the sectors to be numbered in ascending order — with sector 1 followed by sector 2, sector 3, and so on around the circumference of the track. This isn't always the case.

The sector numbering scheme can have a profound influence on disk performance. To understand why, you need to consider how data is actually transferred to and from the drive.

Suppose that you want to save a 10K file to disk. The file is too large to fit in a single sector, so it must be saved in several sectors. Usually, the file is saved in adjacent sectors. The read-write head moves over the appropriate track, reads the sector headers, and looks for the address of the first sector where the file is to be stored. As soon as the sector is located, the first 512 bytes are saved. The system then prepares the next batch of 512 bytes and saves it in the next sector. However, the disk platter is spinning at 3,600 revolutions per minute. If the sectors are numbered sequentially, by the time the read-write head starts looking for the next sector, that sector may already be spinning past the head. If so, the disk must wait for almost one complete revolution before the target sector is once again below the read-write head.

If the system is not fast enough to write two adjacent sectors one after the other, it does not make sense to number the sectors sequentially. The solution is to separate sector 2 from sector 1 by one or more sectors. That way, by the time the system is ready to save the data, the target sector is just rotating under the read-write head.

The term *disk interleave* refers to the sector numbering scheme used on a hard disk. An interleave of 1:1 indicates that the sectors are numbered in sequential order — that is, 3 follows 2, 2 follows 1, etc. Only high-performance systems have a 1:1 interleave, which means that as soon as one sector is written, the system is ready to write the next 512 bytes in the next sector. On a disk with a 2:1 interleave, sector numbers are ordered alternately, as shown in Figure 4-4.

The interleave setting is established during low-level drive formatting. If your drive is configured with the wrong interleave, performance is adversely affected. The drive must rotate an unnecessary number of times to read or write data.

The programs SpinTest and SpinTime, from Gibson Research Corporation, are included on Bob's Shareware Collection disks. Use these programs to report the interleave setting of your hard disk and identify the optimal interleave setting.

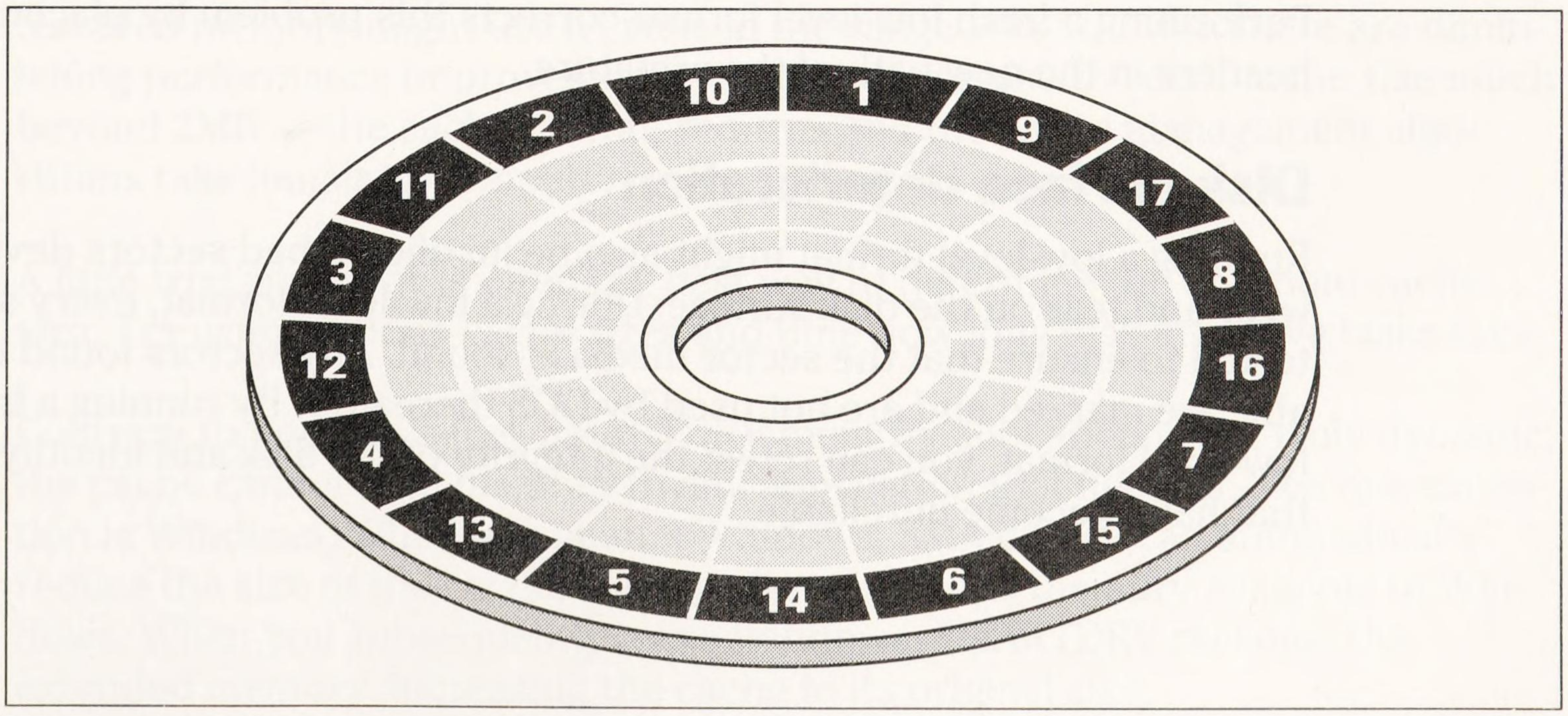


**Figure 4-4:** Interleave settings for a hard disk with 17 sectors per track and a 2:1 interleave.

If you discover that your disk has an inappropriate setting, DOS won't be able to help you. Drives are normally low-level formatted at the factory or by the dealer. DOS, therefore, does not provide any low-level formatting tools. You must use a third-party tool, such as SpinRite, from Gibson Research Corporation. In general, these programs determine the optimum interleave prior to performing a low-level format.

A destructive low-level format will, as the name implies, destroy all the data on the disk; both the sector IDs and the general data area of each sector are rewritten. Some disk drives can be rendered useless (except, perhaps, as a paperweight) if they are destructively formatted because the format will destroy vital drive statistics that are stored on the drive itself. Some of the early SCSI drives, for example, store important data on the disk. SpinRite can perform a nondestructive low-level format, where the data is left intact and only the sector IDs are rewritten. If you do not get the low-level formatting program directly from the drive manufacturer, always use a nondestructive formatting tool like SpinRite.

You may also need to low-level format a drive if you experience frequent error messages saying that sectors were not found. These messages may be caused by drive alignment problems or by disk surface deterioration.

### Drive alignment problems

Like all devices with moving parts, a disk drive is subject to wear. When the sector address information is first recorded on the hard disk, the drive is new and the head armature is accurately positioned above each track. As the disk ages, the armature mechanism can wear slightly, causing the read-write head to shift so that the original sector header is not located directly below the head. When the armature drift is large enough, the sector heads are too far from the head position, and the sector address information cannot be read. This is usually the cause when DOS presents the `Sector not found` message.

Performing a fresh low-level format corrects this problem by placing new sector headers in the new path of the armature.

### Disk surface deterioration

Running a low-level format might also be justified if bad sectors develop due to wear and tear on the disk surface. During a low-level format, every sector is tested to ensure that the sector media is sound. Any sectors found to be unreliable are marked and are not used by DOS thereafter. By running a fresh low-level format, you can retest the integrity of the disk and identify any sectors that have become unreliable.

# Improving throughput with caches

One of the most dramatic ways to improve the performance of a disk is to install a software disk cache. All PC users, in fact, should use a disk cache. DOS's disk cache program is SMARTDRV.

Before DOS 6, the DOS disk cache program was called SMARTDRV.SYS and was always loaded as a device driver in CONFIG.SYS. Now, the disk cache is named SMARTDRV.EXE and is both a device driver and an executable program. If you have an entry in CONFIG.SYS for SMARTDRV.SYS, make sure that you replace it with SMARTDRV.EXE.

NOTE

The DOS 6 SMARTDRV does not support an expanded memory cache, only an extended memory cache.

## Using SMARTDRV

When using SMARTDRV, you need to consider the issues outlined in the following few sections.

The complete syntax for SMARTDRV is detailed in Part IV, "The Complete DOS 6 Command Reference."

### How much memory should you assign to the cache?

We all know that memory is a precious commodity, and few of us are using systems with more than 20MB of RAM (yet). As a general rule, the cache size should be as large as possible. There is, of course, a direct trade-off between the size of the cache and the amount of extended memory available to applications software.

Ideally, the cache size should range from 0.5MB to 2MB, depending on the amount of installed memory. If your system has 4MB or less of extended memory, a cache of 0.5MB to 1MB is appropriate. Systems with 4MB to 8MB of installed memory might use a cache in the range 1MB to 2MB. There are diminishing performance improvements when you increase the disk cache size much beyond 2MB — the cache hit rate reduces and the cache management algorithms take longer to compute.

A little trial and error is often the best way to determine the optimum cache size. Try using various cache sizes and time how long disk-intensive tasks take.

Contrary to popular belief, SMARTDRV memory allocation is not truly dynamic; the cache cannot lend memory to any application that needs it. The one exception is Windows. When you launch Windows, SMARTDRV can automatically reduce the size of the cache, leaving more extended memory available to Windows. When you subsequently leave Windows, SMARTDRV reclaims the extended memory, increasing the cache to its original size.

## How Disk Caches Work

All disk cache software is based on one simple truth: Memory access is a lot faster than disk access. Without an active disk cache, a computer system takes the following steps to process data from disk:

1. The microprocessor asks the hard disk controller for some data.
2. The disk controller determines precisely where on the disk the data is stored.
3. The disk controller moves the read-write head over the appropriate cylinder.
4. Each sector header is read until the target sector is located.
5. The data is read from the sector and passed to the disk controller.
6. The disk controller decodes the data and passes it via the bus to memory.
7. The data is passed from memory to the microprocessor via the bus.

No wonder it seems to take so long. Disk caches use three main techniques to improve file-reading performance:

- **File mirroring:** The first time you read a file into memory, the disk cache software keeps a copy of the file, by sector, in the memory cache. The next time a program tries to access the same file, the disk cache software can intercept the request and pass the file directory from the memory cache, avoiding another time-consuming trip to the disk. Eventually, the cache memory fills up with data. The better cache programs use a sophisticated algorithm to determine which cache data should be overwritten when the cache is full, but basically, they discard the least used sectors.
- **Track buffering:** More often than not, the system reads one block of data in one request and then immediately initiates a second request for the adjacent block of data. The concept behind track buffering is simple: While the system is reading one sector, it might as well read the remaining sectors on the track and keep them in a memory cache. The read-write head is already in position over the next sector, so the overhead of reading the additional sectors is minimal. If a request is made for the next sector, the disk cache software can supply the sector data directly from RAM. Track buffering can speed up all file accesses, even the first time you access a file.
- **Write buffering:** Write buffering is similar in principle to print buffering. The disk cache software stores in a memory cache data that a program is trying to save to disk. The main application program, thinking that the file has been saved, can then proceed with other operations while the disk cache software saves the data to disk in the background. When a system reads a file, the order in which sectors are processed is important, but during file saving, the order in which sectors are written to the disk is not important, provided that all the sectors are saved in a timely fashion. The more advanced disk cache programs analyze the data in the write cache and reorganize the sector-saving sequence so that the data is written in the most efficient way. This strategy reduces read-write head movement and reduces the delays incurred while the system waits for the correct sector to spin beneath the heads. Although this does speed things up, there are risks associated with having unwritten information left in memory; there's more to lose during a crash or power failure, for example.

If your hard drive already has a hardware cache, adding a software cache will not improve your throughput much and will likely slow things down. Try out your favorite programs with and without the cache to see whether you really get an improvement.

Windows, like any disk-intensive program, benefits from a disk cache. However, the less extended memory available to Windows, the more Windows must access the disk. Make the cache too big, and Windows must use the disk more; make the cache too small, and Windows doesn't get much benefit from it. Catch-22 — or maybe Cache-22. As a simple rule of thumb, you should probably reduce the cache size by 50 percent when running under Windows.

When you launch SMARTDRV, specify the normal cache size, followed by the Windows cache size in kilobytes. The following command, for example, creates a 4MB standard cache with a 2MB Windows cache:

```
smartdrv 4096 2048
```

If you don't specify a cache size, SMARTDRV uses a default value based on the total of installed memory. Table 4-2 shows the default cache sizes.

### Do you need double buffering?

The only reason to install SMARTDRV as a device driver in CONFIG.SYS is to enable *double buffering*. Double buffering can resolve conflicts between the disk controller and the memory manager. If your system locks up when you are installing device drivers into upper memory or running Windows in Enhanced Mode, double buffering may provide the solution.

When you enable double buffering, data is moved first from the disk cache to a conventional memory buffer and then off to disk. This technique is needed on some systems that use a bus-mastering disk controller. The problem is most common on older SCSI devices, but some ESDI drives and MCA hard disk controllers may also exhibit the problem. Most systems do not need double buffering.

Fortunately, there is an easy way to determine whether or not you need the double buffering system. Install SMARTDRV.EXE with the /DOUBLE_BUFFER switch in CONFIG.SYS and in AUTOEXEC.BAT. Also make sure that your system is configured to use upper memory; use EMM386 and the DOS=UMB statement.

**Table 4-2 SMARTDRV Default Cache Sizes**

| *Extended Memory* | *DOS Cache Size* | *Windows Cache Size* |
|---|---|---|
| 1MB (or less) | All extended memory | 0K |
| 2MB (or less) | 1MB | 256K |
| 4MB (or less) | 1MB | 512K |
| Less than 6MB | 2MB | 1MB |
| More than 6MB | 2MB | 2MB |

Refer to Chapter 5 for more information on upper memory.

Use your system for a while and then execute SMARTDRV with no switches. A summary of the cache status, similar to the following, is displayed:

```
Microsoft SMARTDrive Disk Cache version 4.1
Copyright 1991,1993 Microsoft Corp.

Cache size: 2,097,152 bytes
Cache size while running Windows: 2,097,152 bytes

                 Disk Caching Status
drive    read cache    write cache    buffering
--------------------------------------------------

  A:        yes            no            no
  B:        yes            no            no
  C:        yes            yes           no
  D:        yes            yes           no
  E:        yes            yes           no

For help, type “Smartdrv /?”.
```

If “no” appears in the buffering column for every drive, you do not need double buffering, and you can remove the SMARTDRV.EXE statement from CONFIG.SYS. However, if any drive has a “yes” entry, you must continue to use double buffering.

If the double buffer column has a “-” entry, there has been insufficient activity for SMARTDRV to assess whether double buffering is needed. Perform some additional tasks on the PC and try again. Eventually, the “-” entry will be replaced by “no” or “yes.”

Do not use the DEVICEHIGH statement with SMARTDRV in CONFIG.SYS; load SMARTDRV low, using DEVICE. It only consumes about 2K of conventional memory.

### Is delayed write caching appropriate?

The reason for installing a disk cache is to improve hard disk performance. Clearly, write caching enhances the performance more than basic read caching does. The only drawback is that chances for data loss increase, because data is not immediately written to disk. However, data loss occurs only if you lose power to your system, press the Reset button, or a program locks up while a file-saving operation is in progress. SMARTDRV is smart enough to intercept a Ctrl-Alt-Del reboot and flush the buffers (save the data) before rebooting.

If you use a third-party utility in a batch file to reboot the system, you might consider adding the following statement to flush the buffers before rebooting:

```
smartdrv /c
```

Because SMARTDRV saves the data within a few seconds, the odds of losing data are not great. If you use a PC in an environment where power failures or program crashes are common, you might consider disabling the delayed write cache. Personally, I like speed and always use write caching.

GEM

## Resolving communications problems with SMARTDRV

If you use SMARTDRV on a system that has DoubleSpace installed, you may experience intermittent problems running communications at 9600 baud and above. The problems may be related to the 8250 UART communications controller, which is unable to handle the interrupt bombardment. You can easily determine your UART type using MSD. If you select the COM Ports option from the main menu, MSD displays (on the bottom line) the UART model.

One quick solution might be to disable the write-caching feature of SMARTDRV. For example, if you are caching drives C and D, specify the drives (without the + or – switch) as follows:

```
smartdrv c d
```

NOTE

If you specify a drive letter with the plus (+) character, SMARTDRV enables read caching and write caching. If you specify the drive letter with a minus (–) character, SMARTDRV disables caching altogether on that drive. However, if you specify the drive without a plus or minus character, read caching is enabled and write caching is disabled.

If it irks you to disable write caching or if it simply doesn't fix the problem, consider upgrading the UART chips to the much more capable 16450 model or, better still, the 16550A model. They cost anywhere from $25 to $35.

## A Brief History of the UART Chip

Each serial port on a PC has a *Universal Asynchronous Receiver/Transmitter* (UART) chip. This chip acts as a translator, converting incoming serial communications into the parallel input that the main PC circuits use. Similarly, it converts parallel data into serial signals for communication to the attached serial device.

The original PC used the 8250 UART chip. Even back then, this chip had a reputation for being slow and awkward. Surprisingly, many of today's PCs still use this antiquated chip (my 33 MHz 486 included). This chip was never designed to accommodate today's high-speed communications. In 1984, manufacturers started to replace the 8250 with the 16450 chip, and then in 1987, the 16550 took the throne. The latest incarnation is the 16550A UART, which includes a tiny, 16-byte buffer that enables it to handle high-speed bursts of data.

## Using SMARTDRV with DoubleSpace drives

To optimize cache efficiency, always configure SMARTDRV (or any other disk cache, for that matter) to cache the host drive and do not cache the DoubleSpace drive. For example, if the host drive is D and the DoubleSpace drive is C, use the following switches to install SMARTDRV:

```
smartdrv e+ c-
```

## Avoiding bad cluster problems

SMARTDRV's track buffering may cause a problem in one situation. When you enable track buffering using default settings, SMARTDRV, after reading a requested cluster, reads the additional clusters on a track in anticipation of a future request for them. SMARTDRV does not analyze the FAT when reading these extra clusters, and on occasion, it tries to read bad clusters — that is, clusters that contain unreliable sectors. Some hard disk controllers deal with this situation gracefully, and others bump and grind for a second or two, but some retry for a significant period of time.

GEM

If you experience problems that may be related to SMARTDRV reading bad clusters, configure SMARTDRV so that track buffering is disabled. Set the read-ahead buffer to 0 by using the /B:0 switch and set the element size equal to your disk's cluster size. Assuming the cluster size is 4,096, the following command resolves the track buffering problem:

```
smartdrv /b:0 /e:4096
```

If you don't know the cluster size, run CHKDSK and review the line *nnnn* `bytes in each allocation unit`. The cluster size is the *nnnn* value.

## Using SMARTDRV with CD-ROMs

Time for the bad news: SMARTDRV cannot cache CD-ROM drives. Ironically, CD-ROMs are slow and desperately need a cache performance boost. But if you want to cache a CD-ROM, you need to use a cache program other than SMARTDRV.

CAUTION

Never cache a DoubleSpace drive. Always cache the host drive — the drive that contains the hidden DoubleSpace volume file DBLSPACE.*nnn*. SMARTDRV has a built-in safety feature that prevents you from caching a DoubleSpace drive, but most third-party cache programs do not.

## Using Fastdisk with Windows

Under normal circumstances, all disk I/O is handled by the BIOS, which can function only in Real Mode. When a Windows program is running in Protected Mode, it must throttle down into Real Mode to perform file I/O and then switch back into Protected Mode to proceed. Needless to say, this takes time.

To improve Windows performance in Enhanced Mode, Microsoft introduced Fastdisk, a form of device driver that reduces the amount of Protected- to Real-Mode switching during file I/O. To be specific, the swap file I/O is performed in Protected Mode, but standard disk I/O still requires a switch into Real Mode. Additional benefits of Fastdisk include the ability to run more DOS-based applications simultaneously and improved performance when a DOS application is running in the background. The benefits of Fastdisk are most apparent on systems with low RAM (5MB or less) and systems that run multiple DOS sessions.

To activate Fastdisk, you must open the Windows Control Panel, select the 386 Enhanced icon, and then select the Virtual Memory button. When the Virtual Memory dialog box is displayed, select Use 32-Bit Disk Access to enable Fastdisk.

Fastdisk is very hardware specific and only functions with controllers that are compliant with the Western Digital 1003 controller interface standard. This standard was selected because the vast majority of IDE drives are compliant. If you use a SCSI system or any drive that uses device drivers to exceed DOS's 1,024 cylinder limit (discussed earlier), you should not use Fastdisk. Many vendors of high-capacity, high-performance drives are supplying their own variations of Fastdisk to provide Protected Mode access. Adaptec, the dominant SCSI force, has not yet released a Fastdisk-style driver for Windows, but other SCSI vendors have.

NOTE

If you are unable to run Windows after activating Fastdisk, you are probably using an incompatible disk controller. If Windows won't even start, edit the SYSTEM.INI file and turn the 32-Bit Disk Access off by using the following command:

```
32bitdiskaccess=off
```

## Using RAM drives

As the name suggests, a RAM drive (or RAM disk) is a disk drive that exists in memory. A RAM drive performs like any other disk drive, with two exceptions:

- A RAM drive is in memory and so is very fast.
- The files stored on a RAM drive are lost when you turn off or reboot the system.

RAM disks gained popularity in the mid- to late 1980s, when many power users had systems equipped with a few megabytes of RAM but had no applications software that could take advantage of it. By using RAMDRIVE.SYS (or its variant, VDISK) to create a RAM disk in extended memory, they could at least use some of the expensive hardware and get a performance boost as a bonus.

RAM drives are fast, but they have too many limitations. In truth, a disk cache using delayed-write buffering is a far more flexible and effective way to use RAM than a RAM drive. With a disk cache, you don't have to copy files from the hard disk to the RAM drive; you don't have to worry about having too little room on the drive; and you don't have to make sure that data is copied back from the RAM drive to the hard disk before rebooting.

To put it bluntly, don't use a RAM drive; use a disk cache instead.

## Using FASTOPEN

FASTOPEN was first introduced in DOS 3.3. It was designed to enhance disk performance by keeping directory information about recently accessed files in a memory buffer. The second time a file is read, DOS can get the file details from RAM rather than having to access the directory and FAT on disk. Although the concept sounds good, tangible performance gains are realized only when you use programs that open and close files frequently, such as database applications.

You realize far greater performance gains if you use SMARTDRV or a third-party disk cache. These programs cache the file data along with the location details. In short, use SMARTDRV instead of FASTOPEN.

## Reviewing the benefits of defragmentation

Not all files are stored neatly in consecutive clusters. When you save a file, DOS tries to save the file in one contiguous block on the unused part of the disk. However, if the disk is near capacity and there isn't room for the entire file, DOS accesses the FAT and starts to save the file in the first available cluster. DOS then checks the FAT again to find the next free cluster and saves the file's second cluster in that space. The process continues until the entire file is saved to disk.

When a file is saved in noncontiguous clusters like this, it is said to be *fragmented.* If you have a disk that is nearly full (whose isn't?), and you regularly delete and create or copy files, there is a good chance that some of your files will be fragmented.

Fragmented files have a negative effect on performance and data recovery. When a file is fragmented, the read-write heads must jump from cluster to cluster to save or load the file. All this head movement takes time. It can take more than twice as long to access a heavily fragmented file than it does to access a file stored in contiguous clusters.

In addition, your chances of recovering a fragmented file that has been accidentally formatted or deleted are greatly reduced. Only when a file is stored in contiguous clusters is there a reasonable chance that you can recover data using UNFORMAT or UNDELETE (when delete sentry or delete tracker are not enabled).

For more information on data recovery and the benefits of defragmentation, refer to Chapter 3.

A quick and easy way to see whether a file is fragmented is to use CHKDSK. If you specify a filename or filemask with the CHKDSK command, DOS reports the overall disk status and lists any files matching the filemask that are fragmented. For example, the following output was generated by issuing the command CHKDSK *.* in the directory E:\DOC:

```
Volume UPSTAIRS-E  created 11-16-1992 1:35p
Volume Serial Number is 1996-B0AA

 167059456 bytes total disk space
  48533504 bytes in 3 hidden files
    598016 bytes in 143 directories
  80965632 bytes in 2035 user files
  36954112 bytes available on disk

      4096 bytes in each allocation unit
     40786 total allocation units on disk
      9022 available allocation units on disk

    655360 total bytes memory
    586000 bytes free

E:\DOC\READ.ZIP Contains 3 non-contiguous blocks
E:\DOC\DS00RAFN.DOC Contains 2 non-contiguous blocks
E:\DOC\DS00RA0Z.DOC Contains 2 non-contiguous blocks
E:\DOC\DS00RA06.DOC Contains 5 non-contiguous blocks
```

## Using DEFRAG

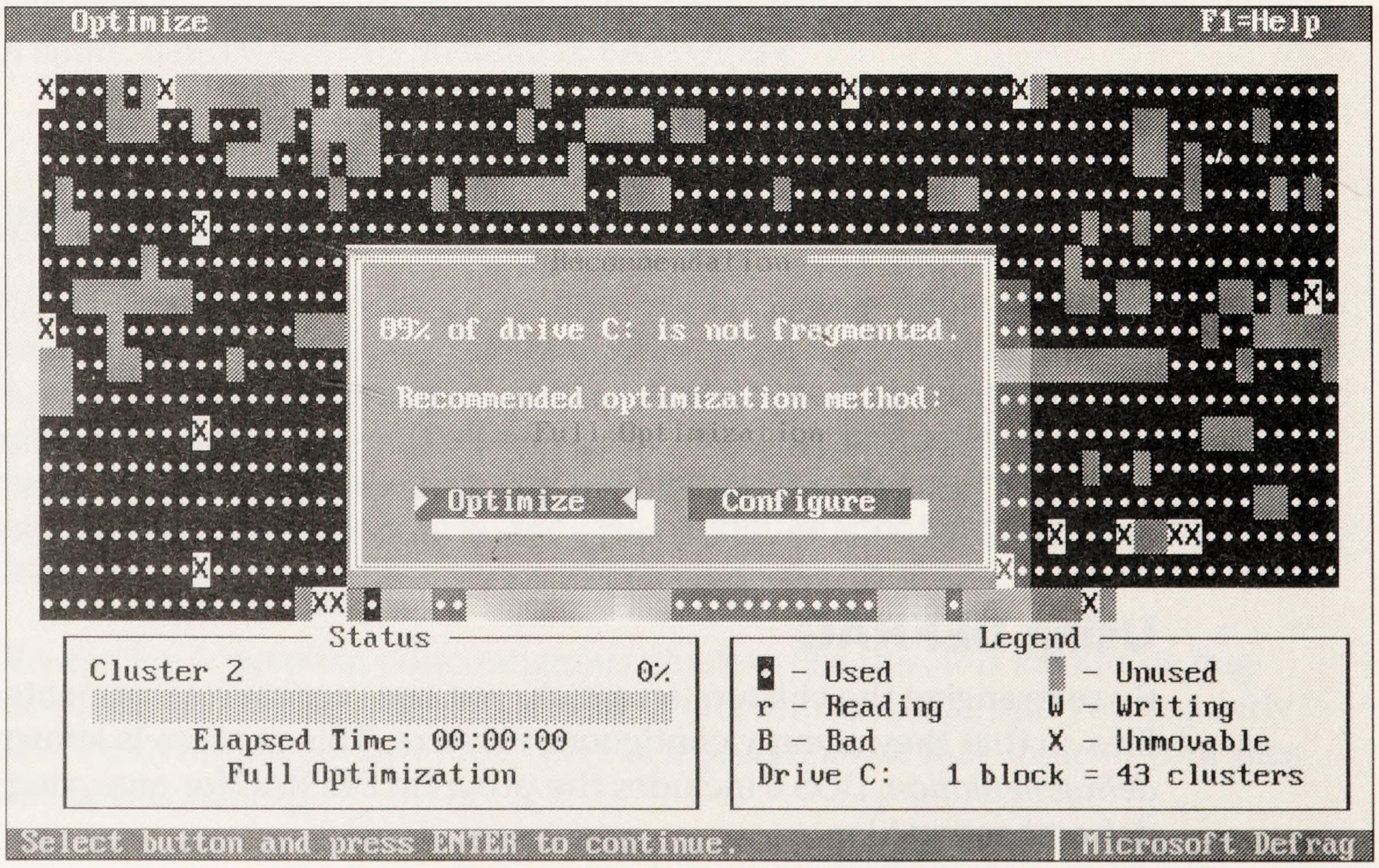
By rearranging the clusters on a disk, you can reorganize fragmented files so that they occupy contiguous clusters. This process is known as *defragmentation.* DOS 6 includes the program DEFRAG for analyzing and defragmenting files.

DEFRAG supports a variety of switches, allowing you to execute the entire defragmentation process from the command line. Using switches, you can specify the optimization method and the sort order for directories, and even instruct DEFRAG to reboot the system when it finishes. The full syntax for DEFRAG is included in Part IV, "The Complete DOS 6 Command Reference."

If you execute DEFRAG with no switches, you can use the program interactively, selecting commands and options from the menu. When DEFRAG is executed interactively, the hard disk is analyzed and DEFRAG reports the most appropriate optimization method (see Figure 4-5).

Defragmentation takes time, so DEFRAG offers two defragmentation methods:

- **Full Optimization:** This method takes the longest amount of time (from 30 to 60 minutes in some cases) but offers the most performance improvements. All directories are moved to the beginning of the disk — in other words, the outermost tracks, near to the FAT. This means that a file's starting cluster can be determined with minimum head movement. All the files are unfragmented and then moved to the outermost part of the disk, leaving the entire free space on the disk in one contiguous free block.
- **Files Only:** This method is a lot faster, but it only defragments the files. Directories are not moved to the beginning of the disk; files are not moved to the outside of the disk; and unused free clusters remain scattered about the disk. When you use this method, new files tend to be fragmented more quickly, as they tend to occupy the noncontiguous clusters.



```
Recommendation

89% of drive C: is not fragmented.

Recommended optimization method:
Full Optimization

Cluster 2                                  0%
Elapsed Time: 00:00:00
Full Optimization

● - Used        ▒ - Unused
r - Reading     W - Writing
B - Bad         X - Unmovable
Drive C:    1 block = 43 clusters

Select button and press ENTER to continue.   | Microsoft Defrag
```

**Figure 4-5:** Running DEFRAG interactively.

DEFRAG displays a form of road map that details the fragmentation of a disk. Compare the disk in Figure 4-6, which was fully optimized, with the one illustrated in Figure 4-7, which was optimized with the Files Only option. Notice all the empty clusters scattered around the disk on the second disk.

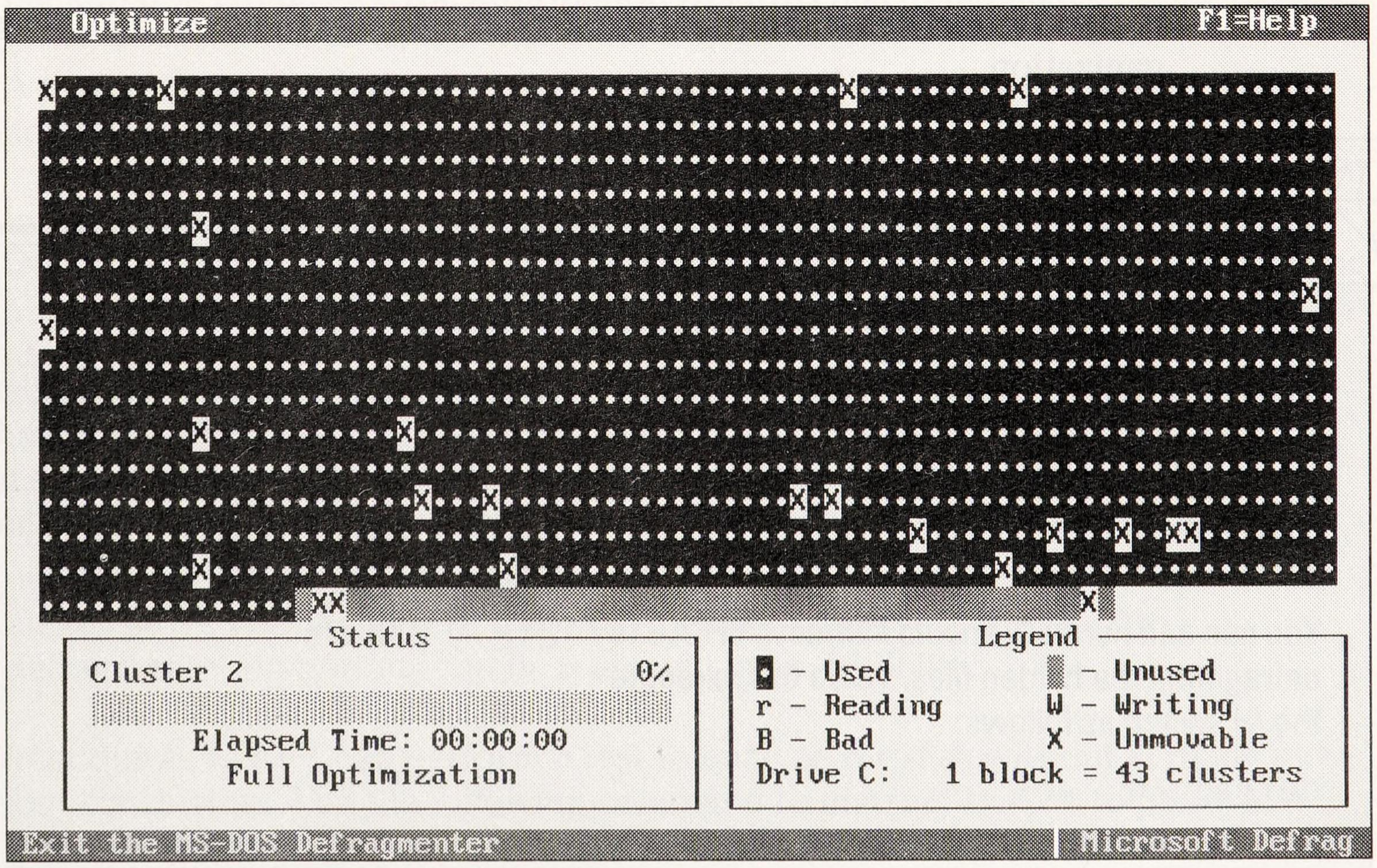
If a large section of the disk map is marked with Xs, it probably indicates a DoubleSpace volume or a Windows swap file. DEFRAG automatically defragments a DoubleSpace drive, and the Windows swap file is always stored in a contiguous block and so doesn't need defragmenting.

During the defragmentation procedure, you can sort directory entries so that files are listed in order by name, extension, date, or size. This doesn't affect the physical location of a file — simply the order that it appears in directory listings.

Because fragmented files impair performance and reduce the chances of error recovery, you should run DEFRAG on your hard disks at least once a week.

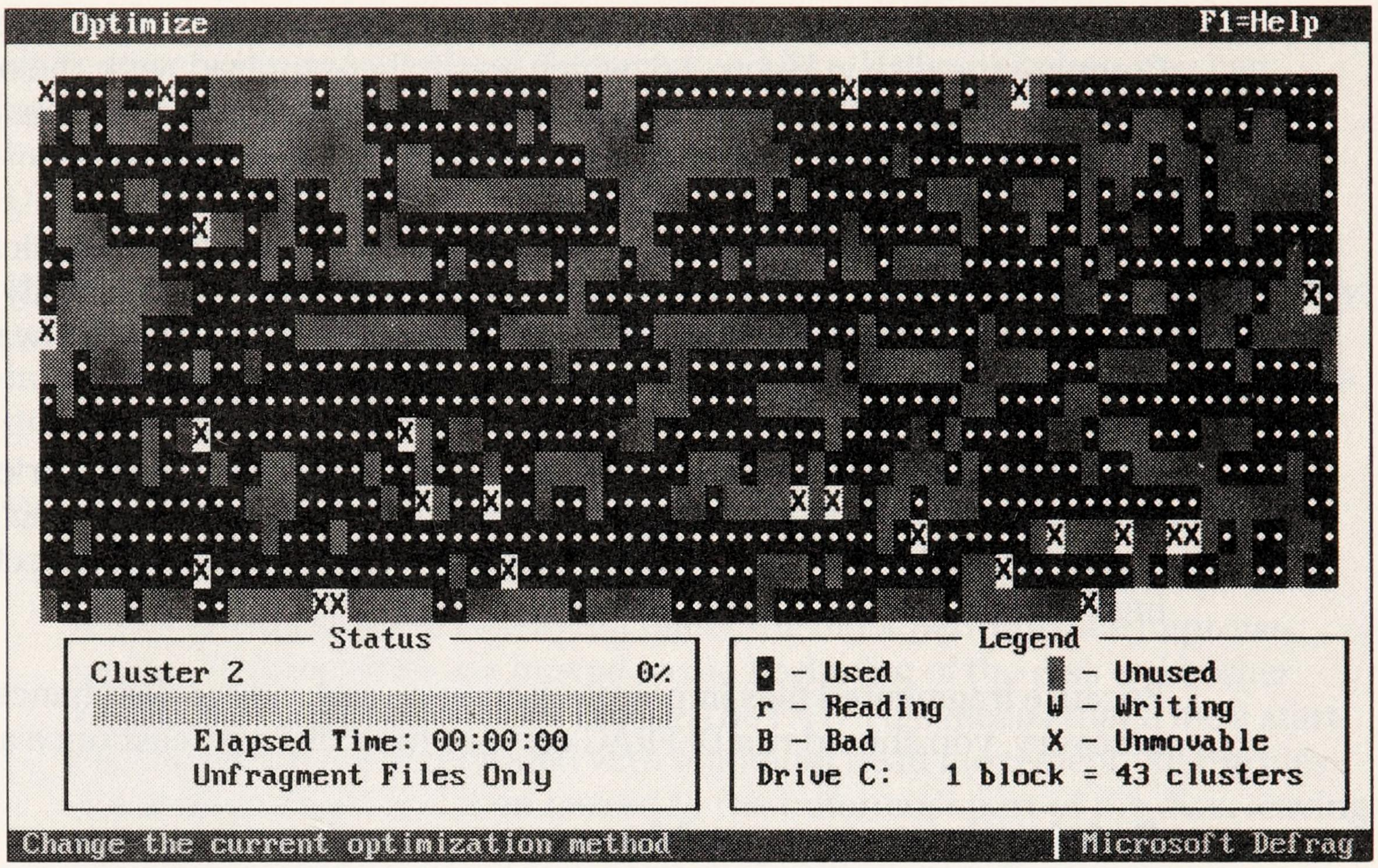
## Solving memory problems

DEFRAG only uses conventional memory to defragment a disk. Sad but true. You may experience insufficient memory problems trying to run DEFRAG on big drives (400MB+) that have large numbers of directories and files.



```
Optimize                                                  F1=Help

Status                              Legend
Cluster 2                     0%    ▪ - Used          ▨ - Unused
                                    r - Reading       W - Writing
Elapsed Time: 00:00:00              B - Bad           X - Unmovable
Full Optimization                   Drive C:    1 block = 43 clusters

Exit the MS-DOS Defragmenter                       | Microsoft Defrag
```

**Figure 4-6:** A disk defragmented using the Full Optimization method.



```
Status
Cluster 2                              0%
        Elapsed Time: 00:00:00
        Unfragment Files Only

Legend
▣ - Used              █ - Unused
r - Reading           W - Writing
B - Bad               X - Unmovable
Drive C:     1 block = 43 clusters
```

**Figure 4-7**: A disk defragmented using the Files Only method.

In case you are interested, DEFRAG consumes 16 bytes for every file on the system and 48 bytes for every directory, plus some room for itself and the two copies of the FAT. What's left is used for buffering the clusters during the reorganization.

GEM

## Defragmenting Hidden Files

By default, DEFRAG does not move hidden files. A hidden file may be part of a copy-protection scheme, and the associated application may not execute if the hidden file's clusters are moved to a different location.

You can and normally should instruct DEFRAG to defragment the hidden files. To do this, execute the /H switch as follows:

```
defrag /h
```

If you suspect that some of your hidden files may be part of a copy-protection scheme, do not use the /H switch. To see a list of all the hidden files on your system, execute the following DIR command:

```
dir \ /ah /s
```

One approach to solving the memory problem is to boot with a configuration that maximizes the amount of free conventional memory. If you have a 386 or better system, try booting with an empty AUTOEXEC.BAT and with the following entries in CONFIG.SYS:

```
dos=high
dos=umb
device=c:\dos\himem.sys
device=c:\dos\emm386.exe noems ia000-b7ff
```

Add the following line if you are using a DoubleSpace drive:

```
devicehigh=c:\dos\dblspace.bin /move
```

When you boot the system, you should have 675K or more of base memory. That's right — more than 640K! This trick uses some of the graphics memory as base memory. Be sure not to run any graphics applications when the system is configured this way.

After you maximize memory, run DEFRAG with the /U switch to force a Files Only optimization. Finally, run DEFRAG a second time with the /F switch to attempt a full optimization.

If this doesn't work, try deleting all the anti-virus CHKLIST.CPS files, using the MWAV program (discussed in the preceding chapter), and run DEFRAG again. As a last resort, you might delete some files and directories (having made a backup first) to reduce the information DEFRAG has to manage.

A more practical solution is to use a third-party defragmenter that can take advantage of expanded or extended memory.

Included on Bob's Shareware Collection disks is an excellent utility called Disk OrGanizer™ (or DOG, for short), from Soft GAM's Software. As well as providing all the functionality of the DOS DEFRAG command, DOG™ has additional power and flexibility. DOG™ can use EMS and can defragment disks with as many as 32,000 files.

Some of the additional switches for DOG™ give you the ability to exclude specific files; to move select files and directories, based on the PATH setting, to the outermost tracks; to physically place files on the disk in the order of creation; and to move specific files to the outermost or innermost tracks. The defragmentation options can be recorded in an ASCII file.

In addition, DOG™ can perform a test to analyze the disk without actually moving any clusters. Use the following command to instruct DOG™ to perform a dry run on drive C:

```
dog c: /test
```

DOG™ provides four different defragmentation methods, each offering a different balance of speed and defragmentation completeness. The /FILL switch invokes the maximum defragmentation option. The following is a sample output from the command DOG C: /FILL:

```
DOG - Disk OrGanizer, Version 3.17 - April 17, 1992
Copyright (C) 1992, G. Allen Morris III,  All rights reserved.

Analyzing disk ...Reading File Allocation Table ...Testing
File Allocation Table ...
Reading directories ...Testing file allocation ...
Drive C: Volume “UPSTAIRS-C” has:
  136 areas of free disk space.
    1 volume label.
   82 null files.
 6495 user files that are contiguous.
   21 user files that are fragmented with 51 extra fragments.
    2 system files that are contiguous.
    2 system files that are fragmented with 6 extra fragments.
  178 user directories that are contiguous.
    1 hidden directory that is contiguous.

Reading order file.Analyzing disk organization. Please wait...
717 clusters need to be moved to organize this disk.
   10 directories need to be packed.
Is it OK to write to the disk?
```

If you press Y, DOG™ defragments the disk; otherwise, it returns you to the DOS prompt.

If you use a third-party defragmenting tool, make sure that you do not try to run it from within Windows or any other multitasking program.

## Bigger, Bigger, Bigger with DoubleSpace

One of the major new features introduced in DOS 6 is DoubleSpace. DoubleSpace is a combination of device drivers and utilities that can almost double the storage capacity of a standard hard drive. If you are forever running out of disk space, DoubleSpace offers a very economical way of increasing your disk capacity without having to buy a new hard disk.

DoubleSpace saves disk space by compressing data before saving to disk. To take advantage of DoubleSpace, you must create at least one DoubleSpace drive. You can create a DoubleSpace drive by using free space on an existing drive or by compressing the data already stored on a drive. For example, DoubleSpace can take 30MB of free space on a drive and create a new drive that can hold about 60MB of files.

All files stored on a DoubleSpace drive are compressed, and the total storage capacity of the drive is almost double that of a conventional drive — hence the name *DoubleSpace.*

You use the command DBLSPACE to create and manage the compressed DoubleSpace drives. The full syntax for DBLSPACE is described in Part IV, "The Complete DOS 6 Command Reference." Although DBLSPACE supports a wide variety of switches, you also can use DBLSPACE in a full-screen, menu-driven mode by executing DBLSPACE with no switches.

**NOTE**

The first time you execute DBLSPACE, DOS runs a setup procedure to install at least one compressed drive. You can use the full-screen DBLSPACE utility only if you have at least one DoubleSpace drive.

## Reviewing DoubleSpace mechanics

A device driver named DBLSPACE.BIN is the brain behind DoubleSpace. When DOS tries to save data to a DoubleSpace drive, DBLSPACE.BIN intercepts the task, compresses the data on the fly, and then saves the compressed data to disk. When that data is read back from the DoubleSpace drive, DBLSPACE.BIN decompresses the data and passes it to DOS in uncompressed form.

The concept of data compression is not new. The basic approach is to analyze the source data and identify repeated sequences or patterns of bytes (or characters). These repeated sequences are then replaced with markers indicating the location of each matching sequence. The repeated sequence is stored only once, and because the marker is smaller than the full character sequence, space is saved.

Some compression techniques provide optimal compression for text files, and others are designed to compress binary program files the most. Recently, compression algorithms were developed to compress video image and sound files. The algorithms used by DoubleSpace are based on methods first introduced by the compression experts Lempel and Ziv. The Lempel-Ziv method employs a general compression algorithm that offers the best overall compression for a mixture of file types — text files, program files, clip art, etc. Some files compress better than others, but as a rule of thumb, DoubleSpace compressed files are half the size of the normal (uncompressed) files.

The primary difference between DoubleSpace and file compression tools such as PKZIP and ARJ (which is included on the enclosed disk) is that DoubleSpace files are compressed in real time. That means that the files are compressed and uncompressed automatically by the device driver; you do not need to manually compress the files or to decompress or extract them before they can be used.

Also, because DoubleSpace must compress and uncompress more quickly than the manual compression utilities, it trades off squeezing that last little bit of compression for additional speed.

## Understanding the compressed volume file

DoubleSpace creates one or more additional drives to store files. Any files that are saved to a DoubleSpace drive are automatically compressed.

Each DoubleSpace drive actually is a large hidden file stored on a normal drive. The file is called a *compressed volume file* (or CVF-) and has the filename DBLSPACE.*nnn,* where *nnn* is a sequence number (DBLSPACE.000, DBLSPACE.001, and so on).

Suppose that your system has two hard disks installed, and you decide to dedicate 20MB of physical drive D to be a DoubleSpace drive. DoubleSpace creates a 20MB hidden file in the root of drive D and also tricks DOS into thinking that there is a new drive — drive J, for example. Drive J appears to have about 40MB of free space. The drive that stores the hidden CVF is said to be the *host* drive. In the example, drive D is the host of drive J.

Whenever you save a file on drive J, DBLSPACE.BIN compresses the file and stores it in the large hidden file on drive D. Similarly, when you load a file from drive J, DBLSPACE.BIN extracts the file from inside the hidden file on drive D and passes it to DOS.

In a way, the CVF is like a giant ZIP or ARJ file that is accessed behind the scenes whenever DOS tries to load a file from the compressed drive.

TECHNICAL STUFF

A CVF file contains all the normal drive structures, including a boot sector, FAT, and root directory. A CVF also includes a sector allocation table that maps standard FAT clusters to DoubleSpace sectors. DBLSPACE.BIN allocates data on a sector-by-sector basis, unlike DOS, which uses clusters. In addition to saving space by compressing data, DoubleSpace sector-based allocation saves space by avoiding the normal waste that occurs when every small file must occupy at least one cluster.

## Compressing an entire drive

When you want to compress an existing drive, files and all, DoubleSpace operates a little differently. For example, consider a system that has only one installed hard disk: drive C.

DoubleSpace creates a CVF in the root directory of drive C and then compresses and transfers all the files from the original drive C to the new CVF. You might expect that after the compression, the original drive C would hold just a few files and the new drive (probably drive H) would contain all the compressed files that used to be on drive C. Not so!

Whenever you compress an existing drive, DoubleSpace automatically *swaps* the drive letters of the physical drive and the compressed drive. So, in this example, the real drive C (the host) appears to be drive H, and the drive that contains all the compressed files appears to be drive C.

Although this drive swapping may seem odd, it is done for a very good reason. After you compress drive C, the newly compressed files still appear to DOS and other programs as if they are on drive C. Imagine the chaos that would ensue if suddenly everything that used to be on drive C were now on drive H. You would need to reconfigure all your programs to run from drive H. It might take days just to edit the Windows INI files.

NOTE

Whenever you create a CVF on drive C, the drive letter of the physical drive C is swapped so that the DoubleSpace drive is assigned the letter C. Even if you create a new CVF using the free space on drive C, DoubleSpace still swaps the drive letter assignments for drive C and the compressed drive.

You can use the switch /LIST with the DBLSPACE command to review the drive letter assignments.

The following output was generated from a system with three physical drives: C, D, and E. The compressed drive I was created using free space on drive D, and the compressed drive J was created using the free space on drive E. Notice that no drive swapping occurred because the DoubleSpace drives were created using free space on existing drives.

```
Drive  Type                        Total Free  Total Size  CVF  Filename
-------------------------------------------------------------------------

  A    Removable drive             No disk in drive
  B    Floppy drive                   0.04 MB     0.69 MB
  C    Local hard drive               7.79 MB   201.97 MB
  D    Local hard drive              29.87 MB   165.11 MB
  E    Local hard drive              14.82 MB   159.32 MB
  F    Available for DoubleSpace
  G    Available for DoubleSpace
  H    Available for DoubleSpace
  I    Compressed hard drive         29.31 MB    37.00 MB  E:\DBLSPACE.001
  J    Compressed hard drive         43.65 MB    43.65 MB  D:\DBLSPACE.001
```

The following output from a DBLSPACE /LIST command was generated on a system with only one physical drive. DoubleSpace was used to compress the drive. Notice that the drive letters have been swapped so that the compressed drive is C and the physical drive is I.

```
Drive Type                          Total Free  Total Size  CVF Filename
-------------------------------------------------------------------------

 A    Removable-media drive         No disk in drive
 C    Compressed hard drive           46.34 MB   139.93 MB  I:\DBLSPACE.000
 D    Available for DoubleSpace
 E    Available for DoubleSpace
 F    Available for DoubleSpace
 G    Available for DoubleSpace
 H    Available for DoubleSpace
 I    Local hard drive                 6.31 MB    80.52 MB
```

NOTE When you compress an existing drive, the CVF filename is always DBLSPACE.000. The 000 extension indicates that the drive letters of the host and the compressed drive should be swapped. When a drive is created from free space on an existing drive, the CVF filename has an extension of 001 or of the next free value if it is not the first CVF on the drive.

## Installing a DoubleSpace drive

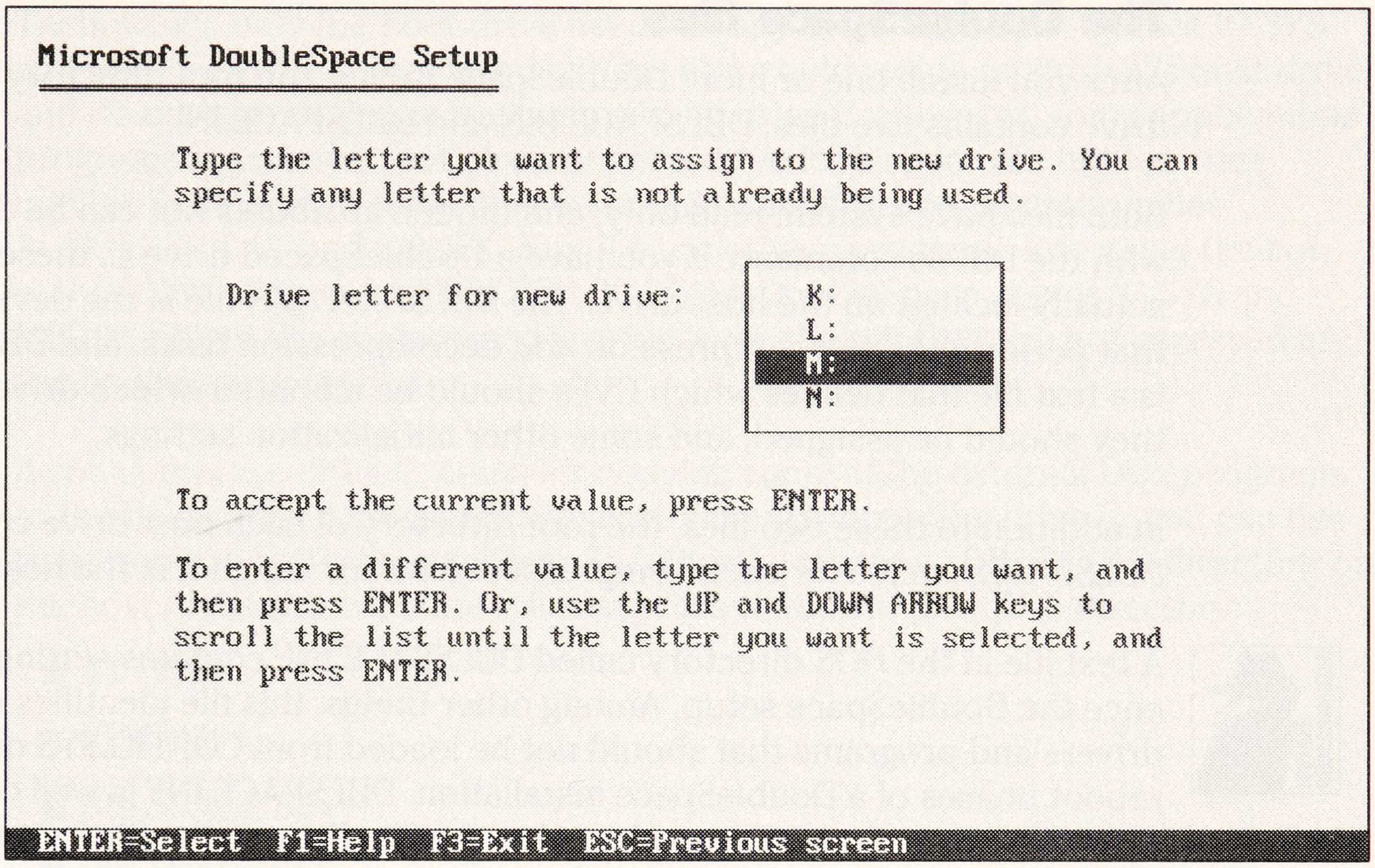
Creating a DoubleSpace drive is a snap. If you are running DBLSPACE for the first time, you are prompted to select either the custom or express setup.

### Express setup

The express option compresses the existing drive C and creates a new host drive, which is usually assigned the letter H. The process involves running CHKDSK on drive C, defragmenting the drive, and creating an empty CVF in the root directory of drive C. In turn, each existing file is read from drive C and compressed. The CVF is then expanded, and the file is written to the CVF. Finally, the originally uncompressed file is deleted. Measures are taken to ensure that there will be no loss of data if the system loses power or is rebooted during the compression procedure. The entire process can take several hours.

### Custom setup

The custom option allows you to select which drive you want to compress and identify whether you want to create a new empty drive or compress an existing drive. You also can select the drive letter that you want to assign to the new drive (see Figure 4-8) as well as the amount of free space you want to leave on the host drive.



```
Microsoft DoubleSpace Setup

        Type the letter you want to assign to the new drive. You can
        specify any letter that is not already being used.

           Drive letter for new drive:        K:
                                              L:
                                              M:
                                              N:

        To accept the current value, press ENTER.

        To enter a different value, type the letter you want, and
        then press ENTER. Or, use the UP and DOWN ARROW keys to
        scroll the list until the letter you want is selected, and
        then press ENTER.

ENTER=Select  F1=Help  F3=Exit  ESC=Previous screen
```

**Figure 4-8:** Custom setup allows you to select the drive letter for the new drive.

Choose the custom setup if you want to compress any drive other than C. You also should use the custom option if you want to compress drive C but control the drive letter of the new host for C, or if you want to specify the amount of free space to be left on the host.

If your drive was already compressed using Stacker (a third-party compression tool), DoubleSpace recognizes that fact and automatically converts the Stacker file to DoubleSpace format. Don't expect the conversion to be fast; it usually takes several hours.

**CAUTION**

If you install a DoubleSpace drive, you are not able to restore your previous version of DOS using the UNINSTALL command. You may want to run DOS 6 for a few sessions before installing DBLSPACE. When you are confident that you want to keep DOS 6 (and I think you will), you can go ahead and create a DoubleSpace drive. While you are at it, you might as well save some more disk space by deleting the OLD_DOS.1 directory — a quick way to do this is by executing the DELOLDOS command. This directory is used by UNINSTALL to restore the previous version of the operating system. Because UNINSTALL won't work, there is no need to keep the OLD_DOS.1 files.

**GEM**

Although DBLSPACE works on huge drives (those larger than 1 gigabyte), the maximum size of a compressed drive is 512MB. Assuming an approximate 2 to 1 compression ratio, this means that you should not use more than 250MB of drive space to create a CVF.

## The DoubleSpace files

After you install one or more DoubleSpace drives, the root directory of the boot drive contains two files: DBLSPACE.BIN and DBLSPACE.INI.

Both files have system, read-only, and hidden attributes but can be displayed with the DIR /A command. If you have a DoubleSpaced drive C, these files are actually located on the host drive. The DBLSPACE.BIN file is the device driver that performs all the compression and decompression tasks, and DBLSPACE.INI is a text file that defines which CVFs should be mounted, which drive letters they should be assigned, and some other initialization settings.

In addition to these two files, the root directory of each host drive contains one DBLSPACE.*nnn* file for each compressed drive for which it is the host.

UNDOCUMENTED FEATURE

A text file in the DOS directory called DBLSPACE.INF contains settings that influence the DoubleSpace setup. Among other things, this file identifies device drivers and programs that should not be loaded from CONFIG.SYS during the reboot phases of a DoubleSpace installation. DBLSPACE.INF is well commented, and each section is clearly explained in the file. Browse the file for further information.

## The DoubleSpace booting process

Although DBLSPACE.BIN is the device driver for compressed drives, you will not find a DEVICE= entry in CONFIG.SYS to load DBLSPACE.BIN, for one very good reason. If CONFIG.SYS itself is stored on a compressed drive, DOS must load the device driver before it is able to read CONFIG.SYS. To work around this problem, IO.SYS searches the root directory of the boot drive looking for the hidden file DBLSPACE.BIN. If it finds the file, it is installed as a device driver. DBLSPACE.BIN is initially loaded at the top of memory, adjacent to the memory area occupied by IO.SYS. After the device driver is installed, the CONFIG.SYS file can be read, and booting proceeds normally.

The drawback to this approach is that the device driver cannot be loaded directly into upper memory because the UMBs are not available so early in the booting cycle. DOS provides another DoubleSpace device driver, called DBLSPACE.SYS, whose sole purpose is to transfer the DBLSPACE.BIN device driver from base memory to upper memory.

DBLSPACE.SYS is not a device driver for DoubleSpace drives; it is a device driver to move the DBLSPACE.BIN device driver into upper memory.

To place both DBLSPACE.SYS and DBLSPACE.BIN into upper memory, use the following command in CONFIG.SYS:

```
devicehigh=c:\dos\dblspace.sys /move
```

If CONFIG.SYS does not contain a DBLSPACE.SYS driver, DBLSPACE.BIN is moved to the bottom of base memory after all the CONFIG.SYS statements are processed.

Technically, only the boot drive needs a copy of the DOS start-up files IO.SYS and MSDOS.SYS. In other words, these files could reside on drive C's host drive and do not need to be on the compressed drive C. However, some applications programs search the root directory of a drive looking for the hidden start-up files to determine which drive should contain the configuration files AUTOEXEC.BAT and CONFIG.SYS. To trick these programs into using the correct CONFIG.SYS and AUTOEXEC.BAT files, DoubleSpace copies IO.SYS and MSDOS.SYS to the compressed C drive, even though they are not accessed in the booting process.

If you compress drive C, consider copying some of the external DOS programs to the host drive. If you ever have any difficulty mounting drive C, you can use these tools to solve the problem. Specifically, create a \DBLTOOLS directory on the host drive and copy the following files to it from the \DOS directory:

- ATTRIB
- CHKDSK
- DBLSPACE.BIN
- DBLSPACE.SYS
- DBLSPACE.EXE
- DBLSPACE.HLP
- DEFRAG.EXE
- DEFRAG.HLP
- EDIT.COM
- EDIT.HLP
- QBASIC.EXE
- MSD.EXE

## Problems with swap files

To improve performance, Windows bypasses DOS and the DoubleSpace device driver when accessing swap files. Swap files, therefore, cannot be located on a compressed drive. When creating a compressed drive, DoubleSpace automatically identifies the presence of a Windows swap file and makes sure that it is situated on the host drive.

If you get a corrupted swap file message from Windows soon after you compress an existing drive, it is probably because the swap file now resides on the host drive and Windows is looking on the compressed file. For example, Windows may be looking at drive C, when the swap file is actually on drive I. You can remove the old swap file and create a new one using the Virtual Memory option in the 386 Enhanced dialog box, accessed via the Control Panel.

You may think that a quick solution to the situation just described is to edit the SYSTEM.INI file and change the drive letter in the `PermSwapDOSDrive` entry, but this only solves part of the problem. You would also need to modify the binary file SPART.PAR (using DEBUG or a programmer's editor) to change the drive entry for 386SPART.PAR. It's far easier to let Windows do the work.

If you plan to install Windows after compressing a drive, be sure to leave enough space on the host (10MB or so) for a swap file. By default, DoubleSpace does not leave sufficient room on the uncompressed drive.

## Drive-mounting problems caused by file fragments

If you get an error message such as `DoubleSpace cannot mount drive n because of an unrecognized error (assertion 105)`, the problem is probably too many fragmented files on the CVF. To correct the problem, you must edit the DBLSPACE.INI file in the root directory of the boot drive and add or modify the `maxfilefragments` entry, using the following procedure:

1. Change directories to the root of the boot drive (the drive containing DBLSPACE.BIN).
2. Change the attributes of DBLSPACE.INI using the following command:

   ```
   attrib -s -r -h dblspace.ini
   ```

3. Edit the DBLSPACE.INI file and make sure that the following line exists:

   ```
   maxfilefragments=1000
   ```

4. Change the file attributes using the following command:

   ```
   attrib +s +r +h dblspace.ini
   ```

5. Reboot the system and, when the CVF is successfully mounted, run DBLSPACE and defragment the file.
6. Edit the DBLSPACE.INI file once more (changing the attributes before and after) and set the `maxfilefragments` entry back to 115.

## How to defragment DoubleSpace volumes

DEFRAG does not normally move a DoubleSpace compressed volume file (CVF) on a host drive because the file is marked as a system and hidden file. If the CVF is located in the middle of the disk, you might want to trick DEFRAG into including the CVFs in the optimization process. Doing so improves the overall optimization.

A DoubleSpace volume is a hidden file containing all the files and directories that appear to be stored on the DoubleSpace drive. The files are stored in the root directory of the host drive. Filenames follow the convention DBLSPACE.*nnn,* where *nnn* is 000, or 001, or 002, etc.

To include a CVF in the optimization process, you must remove the system attribute from the CVF file, run DEFRAG with the /H option, and then reboot the system. The following batch file defragments the host drive I using this technique:

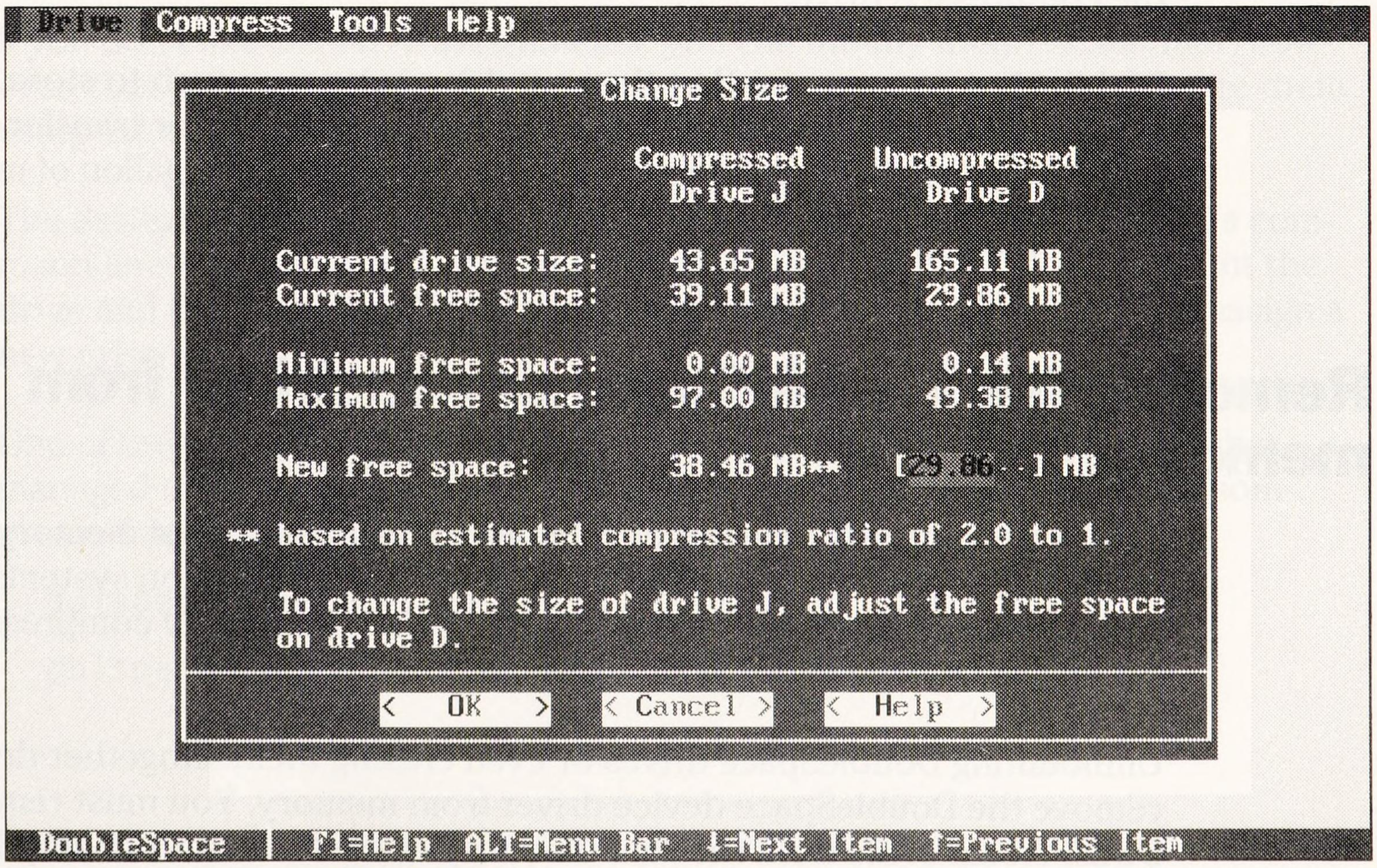
```
@echo off
attrib -s dblspace.000
defrag i: /h /b
```

## Resizing a DoubleSpace drive

Using DBLSPACE, you can easily increase or decrease the size (capacity) of a DoubleSpace drive. Figure 4-9 shows the dialog box in which you change the size of a DoubleSpace drive. The dialog box is displayed when you choose **D**rive⇨Change-**S**ize from the main menu.

The only editable field in the dialog box is the New free space field for the Uncompressed or host drive. The default value is the current amount of free space on the host. To increase the size of the DoubleSpace drive, make the amount of free space smaller than the default. The size of the Double Space drive increases by approximately 2MB for every 1MB you reduce the free space.

If you want to decrease the size of the DoubleSpace drive and increase the free space on the host drive, defragment the DoubleSpace drive before resizing. After you defragment the drive, enter a value larger than the default. Note that


Drive Compress Tools Help
Change Size
< OK >
< Cancel >
< Help >
DoubleSpace | F1=Help ALT=Menu Bar ↓=Next Item ↑=Previous Item

```
                         Compressed     Uncompressed
                          Drive J          Drive D

  Current drive size:     43.65 MB        165.11 MB
  Current free space:     39.11 MB         29.86 MB

  Minimum free space:      0.00 MB          0.14 MB
  Maximum free space:     97.00 MB         49.38 MB

  New free space:         38.46 MB**      [29.86  ] MB

** based on estimated compression ratio of 2.0 to 1.

  To change the size of drive J, adjust the free space
  on drive D.
```

**Figure 4-9:** The Change Size dialog box.

the value cannot exceed the size listed in the Maximum free space field in the Uncompressed column. You cannot reduce the DoubleSpace drive by more than this amount because of the number of files stored on the compressed drive.

## Removing a DoubleSpace drive

You probably will be disappointed to learn that there is no convenient way to remove a DoubleSpace drive. Well, to be accurate, there is no convenient way to remove a DoubleSpace drive without losing all files stored on the drive. Although DoubleSpace can take all files on an existing drive and compress them, creating a new compressed drive, it cannot uncompress a drive and return the files to the original (host) drive.

CAUTION

If you select the **D**rive⇨**D**elete option from the DBLSPACE menu, the drive is deleted, as are all files stored on the drive.

One way to uncompress a drive is to back up all files on the compressed drive, delete the drive using DBLSPACE, and then restore all files from backup to the original drive.

Alternatively, you can copy as many directories as will fit onto the host, delete the directories from the compressed drive, and then shrink the size of the compressed drive as much as possible. By shrinking the compressed volume, you create more space on the host drive. Keep repeating the copy-delete-shrink operation until all the files are transferred to the host drive. You can then delete the compressed drive.

Both techniques assume that the host drive is large enough to store all the compressed drive files. If not, you must delete some files or transfer them to a different drive. As my mother would say, “You can’t get a gallon of milk into a pint jug.”

## Removing the DoubleSpace device driver from memory

The DoubleSpace device driver consumes a valuable 41K of memory. Note that the memory consumed does not vary. In other compression systems, the amount of memory consumed is affected by the number of compressed drives, but this is not true with the DoubleSpace system.

Unmounting DoubleSpace drives or even erasing them altogether does not remove the DoubleSpace device driver from memory. You must remove the DBLSPACE.BIN file from the root directory of the boot drive to stop IO.SYS from automatically loading it into memory.

Because a copy of DBLSPACE.BIN is stored in the DOS directory, you can simply delete the file in the root directory. The file's attributes are set to read-only, hidden, and system, so if you want to use the DEL command, you should first change the attributes using ATTRIB.

Alternatively, you can delete the file using DELTREE — yes, DELTREE. If you specify a particular file rather than a directory, DELTREE can delete the file, regardless of the file attributes. To delete the two primary DoubleSpace files and stop them from being loaded into memory, you can use the following two commands (assuming that the boot drive is H):

```
deltree h:\dblspace.bin

deltree h:\dblspace.ini
```

Remove DoubleSpace from memory in this way only after you delete any DoubleSpace volumes.

## Changing drive letter assignments

To most users (including me) the way DoubleSpace assigns drives seems odd.

If you have only one hard disk installed, DoubleSpace assigns the letter H to the host. If you have two drives installed (C and D), DoubleSpace assigns the letter I to the host of the first drive to be compressed and H to the second. If you have three drives installed, the first host drive is assigned the drive letter J, and so on.

If you don't like the drive assignments, you can modify them. Remember, however, that you can always specify the desired drives when you first create them using the custom setup (see Figure 4-8, discussed earlier).

The easiest way to change drive letter assignments is to use DBLSPACE's command-line switches. The basic approach is to use /UNMOUNT to unmount the drive and then use /MOUNT and /NEWDRIVE to remount the drive and assign a new drive letter.

One of the example configurations discussed earlier had a host drive E that managed the compressed drive I. The following commands change the compressed drive letter to G:

```
dblspace /unmount i:

dblspace /mount=001 e: /newdrive=G:
```

NOTE The =001 switch is used with /Mount because the CVF filename to be mounted as drive G is DBLSPACE.001.

Behind the scenes, DoubleSpace modifies the DBLSPACE.INI file in the root directory of the boot drive. In this example, the INI file had the following contents:

```
MaxRemovableDrives=2
FirstDrive=F
LastDrive=J
MaxFileFragments=128
ActivateDrive=I,E1
ActivateDrive=J,D1
```

After the unmount and remount procedure, the INI file changed to the following:

```
MaxRemovableDrives=2
FirstDrive=F
LastDrive=J
MaxFileFragments=128
ActivateDrive=G,E1
ActivateDrive=J,D1
```

If you want to change the host drive letter (rather than the DoubleSpace compressed drive letter), you need to use a different tactic. You must manually edit the DBLSPACE.INI file. Consider a system in which the C drive is compressed and the host drive is I. The contents of the DBLSPACE.INI file will be similar to the following:

```
MaxRemovableDrives=2
FirstDrive=D
LastDrive=I
MaxFileFragments=115
ActivateDrive=I,C0
```

The following command changes the file attributes:

```
attrib -h -s -r dblspace.ini
```

You can then edit the file and change the ActivateDrive line that contains the C0 entry to the desired drive. In this example, the following INI file contents change the drive to D:

```
MaxRemovableDrives=2
FirstDrive=D
LastDrive=I
MaxFileFragments=115
ActivateDrive=D,C0
```

You could use the following command to set the file attributes back and then reboot:

```
attrib +h +s +r dblspace.ini
```

The host drive changes to D, and the compressed drive stays as C.

## Understanding drive compression ratios

The amount a file is compressed, referred to as the *compression ratio,* depends on the specific file contents. A BMP file, for example, can be compressed a lot more than an EXE file, and a ZIP file can hardly be compressed at all.

On average, the compression ratio is about 2:1, indicating that when a file is compressed, it occupies about half the number of bytes as it did originally. You can use the new /C switch with the DIR command to look at the actual compression ratio of files stored on a compressed drive. The following listing illustrates the output from a DIR /C command:

```
 Volume in drive G is COMPRESSED
 Volume Serial Number is 1BE8-3A0F
 Directory of G:\

CMFILER  ZIP    155953 02-14-93   8:54p   1.0 to 1.0
CMFILER  DOC    203348 11-12-92   5:45a   1.7 to 1.0
DUNES    BMP    308278 05-26-92   1:00a   1.5 to 1.0
CUSTOMER DBF   1660706 11-27-92   3:56p   3.8 to 1.0
MAIN123W EXE   1687360 11-02-92   9:25a   1.4 to 1.0
                  1.9 to 1.0 average compression ratio
        6 file(s)    4015645 bytes
                    25968640 bytes free
```

DoubleSpace maintains an average compression ratio for each compressed drive. At boot time, DOS automatically adjusts each DoubleSpace drive's compression ratio based on the compression ratio of each drive's existing files.

The amount of data that can be stored on a compressed drive depends on the type of files that are to be stored on the drive. For example, there might be room for 10MB of database files (because they compress so well) but only 2.5MB for ZIP files. To provide an average figure, DOS reports the amount of free space available on a DoubleSpace drive by multiplying the physical free space by the compression ratio.

You can use DBLSPACE to modify the compression ratio temporarily. By selecting the **D**rive⇨Change-**R**atio option from the DBLSPACE menu, you can alter the ratio to any value in the range 1.3 to 4.5. However, if you use the command switch /RATIO, you can change the ratio to any value in the range 1.0 to 16.0.

You should realize that modifying the compression ratio doesn't actually change the amount of free space; it simply changes the amount of free space that is reported by DOS. You may want to do this in a few rare circumstances. Suppose that you want to install a road map program onto a compressed drive, and the current drive compression ratio is low (say 1.5) because the drive is used to store a graphics application loaded with large executables and BMP files. When you try to install the new map program, the installation aborts, and a message appears stating that you need 10MB of free space and only 8MB are available. However, you know that in reality, DoubleSpace will have plenty of room because the map data files can be highly compressed. The solution is to doctor the compression ratio to something in the neighborhood of 4:1 so that DOS reports more than 10MB free space to the map installation program.

## Caching compressed drives with SMARTDRV

SMARTDRV is tightly integrated with DoubleSpace. When caching a host drive, SMARTDRV automatically caches each compressed drive supported by the host. What's more, SMARTDRV caches the compressed clusters rather than the standard (uncompressed) clusters. This means that a 1MB cache can effectively cache 2MB worth of data. Smart.

## Compressing data on floppies

The concept of compressing data on floppies has real merit; being able to store nearly 3MB of data on a 3½-inch floppy is very appealing. Unfortunately, the way that DoubleSpace handles compressed floppies is far from graceful. The two major problems follow:

- Even if you don't have a compressed hard drive, the entire DoubleSpace device driver must be installed at boot time to handle compressed floppies. That's 41K of RAM. To make a situation worse, there is no convenient way to use the multiple-boot facility to optionally load DoubleSpace from a menu. You have to manipulate the hidden file DBLSPACE.BIN as discussed earlier.
- To access data on a compressed drive, you must insert the diskette in a drive and then use DBLSPACE to mount the diskette. DOS cannot automatically mount a DoubleSpace drive. In other words, you can't simply put the disk in the drive and start using it. Worse still, every time you change compressed diskettes, you must mount the newly inserted diskette.

NOTE

If you try to access a compressed diskette without first mounting it, all you will see is a single file called READTHIS.TXT. If you type or browse the file, you will see that it contains a message informing you that the drive must be mounted using DBLSPACE.

If you already have compressed drives and so would be loading DBLSPACE.BIN anyway, and you don't mind mounting each floppy, you probably will find compressed floppies valuable. I think I'll wait for the next release!

## *Summary*

This chapter covered all the different ways to get the best performance and capacity from your hard disks. You learned the following points:

- A disk comprises a stack of metal plates coated with metal oxide and a set of read-write heads that transfer data to and from the platter surfaces.
- The data stored on a disk is organized into tracks, sectors, and clusters. A vertical column of aligned tracks is known as a cylinder.
- DOS uses a combination of a directory entry and the FAT to locate the clusters occupied by a file. You can verify the integrity of the data stored in these areas using the DOS command CHKDSK.
- A disk's interleave setting controls the order in which sectors are numbered around the track. Using the optimum interleave setting can have a profound impact on disk performance.
- The DOS disk cache SMARTDRV provides the best way to use RAM to increase disk performance.
- A file is fragmented when its data is not stored in adjacent clusters. Defragmenting files improves performance and enhances the chances of recovering from an accidental format or file deletion.
- You can almost double the capacity of your hard disks by using DoubleSpace. The DoubleSpace device driver DBLSPACE.BIN compresses files when they are saved to disk and decompresses (or expands) them when they are read from disk.
- DoubleSpace can compress existing drives or use the free space on a drive to create a new drive. As a safety measure, you should copy the key DoubleSpace files and ATTRIB.EXE onto the host drive when compressing the boot drive.
- To remove DoubleSpace from memory, you must delete the hidden file DBLSPACE.BIN from the root directory of the boot drive.
- Although DoubleSpace is an excellent hard disk utility, it is cumbersome to use on floppy disks.

The next chapter explains how to improve your system performance even more by making the best use of memory. The chapter provides information about different memory types as well as instructions on how to use the new DOS memory management tools.

# Chapter 5

# Maximizing Memory

---

### *In This Chapter*

- Memory terms, types, and tidbits
- The DOS 6 memory arsenal
- How to check your memory settings with MEM
- The big memory gainers
- From RAM Cram to UMB Cram
- How to load DBLSPACE.BIN high
- How to save every drop of base memory, maximize extended memory, and manage expanded memory
- Ways to solve memory conflicts

---

## Photographs and Memories

When I was a boy scout, I took some photographs at our annual camp and so did another scout's father. A few weeks later, all the scouts were looking at the photographs. Mine were appalling; the other man's were superb. I was mortified. I told the man that he was very talented to master such a difficult subject, and his response was simple yet profound: "It's easy once it's explained to you."

That man later gave me a lesson in photography. He taught me that the shutter speed controls how long the film is exposed to the light, and the aperture controls how much light gets through at any given time. To take good photographs, you simply assess the available light and adjust the shutter speed and aperture accordingly. He was right; it is easy once it's explained!

The same is true for computer memory. At first, the subject can seem daunting, and for good reason. You must deal with DRAMs, SIMMs, RAM, ROM, UMB, HMA, and other CMAs (confusing memory acronyms). But after someone explains the concepts and terms involved, you'll find it hard to imagine why you were so confused.

There's one other parallel between photography and memory. Many people know how to snap a picture but don't know enough about how the camera works to use it to its full potential. Similarly, many people have a working knowledge of memory but don't understand it well enough to use their computers to their best advantage. To squeeze the most from your system, you must learn to make optimum use of all your computer's memory.

In this chapter, you learn the basics of memory — what it is, what it does, and why you can't compute without plenty of it. You also learn more advanced memory concepts that enable you to push your computer to its maximum potential.

## The Memory Crunch

All programs need computer memory to run. An operating system needs memory, a pop-up calendar needs memory, a disk cache needs memory, a mouse driver needs memory — you get the idea. In general, the more functional and feature-laden the program, the more memory it consumes.

All computers have a basic, 640K memory allotment, which is referred to as *conventional* or *base* memory. Many applications can use this standard memory only. Such programs cannot utilize anything but the base memory even if your system has 16MB of memory. The problem with this is that by the time you load start-up programs such as the operating system (COMMAND.COM), mouse and network drivers, and a disk cache, conventional memory may be reduced to 550K or less. (The good news is that DOS 6 provides ways to load many of these start-up programs in memory outside of the 640K memory area, thereby leaving as much precious conventional memory available as possible.)

It is a common misconception to assume that the more memory you install in your system, the more programs you can run. Although there is no doubt that having more memory is desirable, you cannot assume that your programs can utilize all the memory in your computer system. All memory is not alike, and memory can be configured to operate and behave in different ways. For example, on 286 and later systems, you usually can configure memory as *expanded* or *extended memory.* Some programs can use only expanded memory, some can use only extended memory, and some can use either. Lotus 1-2-3 Release 2.4 uses expanded memory, for instance, whereas Lotus 1-2-3 Release 3.4 uses extended memory.

You can use many different techniques to make sure that every nook and cranny of free memory is put to good use. Before exploring these techniques in detail, however, you need to understand some basic memory concepts.

## Memory Terms, Types, and Tidbits

If you are already familiar with memory hardware, memory addressing schemes, and the major memory types (base, upper, high, expanded, and extended), you may want to skip to the next major section, "How to Maximize Memory with DOS 6." Otherwise, I suggest that you read this section because it explains important memory principles that will help you to leverage the most potential from your system.

To fully understand today's confusing memory melee, you need to take a look at the history of the PC.

Since the introduction of the first PC in the early '80s, users (and programs) yearned for more and more memory. People resolved memory shortages by buying more memory chips and adding them to their PCs but soon reached a fundamental design limit — the 640K ceiling. The original 8088/8086 computers, such as the IBM PC and the IBM PC-XT, were designed to provide a maximum of 640K of read/write memory.

In the drive to break through this barrier, vendors formed alliances (such as the Lotus-Intel-Microsoft committee), working together to find ways to go beyond 640K. At about the same time, manufacturers introduced new chips that could address up to 16MB of memory. Standards started to dissolve. The memory techniques used on XT computers often were not appropriate or necessary on AT-class computers. When 386 computers were introduced, further memory options surfaced.

Over the years, a lack of standards and market drive led to a melting pot of terms and techniques.

## ROM vs. RAM

The two most basic classifications of memory are *read-only memory* (ROM) and *random-access memory* (RAM).

When you first turn on your PC or reboot the system, the PC goes through a special initialization process. You may have noticed the memory check being displayed or the disk drives spinning before DOS is loaded. The instructions that guide this process are stored in ROM. As its name implies, read-only memory is read but never modified. The instructions are fused into the memory chips during manufacturing and are permanent.

By contrast, random-access memory, or RAM, is initially empty (at least, it contains no meaningful data). Your computer uses RAM to store programs and data temporarily. RAM is the lifeblood of computer programs.

ROM is similar to a book, in which the text is preprinted and cannot be changed. RAM, on the other hand, is like a chalkboard onto which data can be written, erased, and rewritten.

As a PC user, your interest lies in gaining effective use of RAM. The instructions stored in ROM are indispensable, and your computer won't work without them, but in general, you do not need to understand or configure ROM. Most of your energy will be focused on optimizing RAM. Some utilities even convert unused ROM memory addresses to trick programs into thinking ROM is RAM. More about that later.

## ROM-BIOS

In addition to start-up ROM, which controls the booting process, there is ROM-BIOS, which stands for *read-only memory — basic input/output system.* DOS uses ROM-BIOS to perform low-level tasks such as retrieving a sector of data from the disk or outputting data to a COM port. ROM-BIOS is therefore very hardware dependent.

If you add new hardware to an older system — for example, if you add a new hard drive to a Compaq Deskpro 286 — you may find that you also need to replace the ROM-BIOS chips. These memory chips are not soldered into the circuit board; you can simply pull out the old chips and push in the new ones.

GEM

The hidden DOS file IO.SYS, located in the root directory of the boot disk, includes patches, updates, and extensions to ROM-BIOS. This file provides a convenient mechanism for extending a PC's BIOS capabilities without replacing the ROM-BIOS chips.

When the first IBM PCs were distributed, they included proprietary ROM-BIOS code developed by IBM. Following the emergence of non-IBM PCs, known as *clones* or *PC compatibles,* a number of vendors began developing their own versions of ROM-BIOS. Some of the best known BIOS developers are American Megatrends, Award, and Phoenix Technologies.

Every ROM-BIOS includes the manufacturer's name, a version number, and the date when the BIOS was written. Use the DOS 6 utility MSD.EXE to review the type of ROM-BIOS installed on your PC.

## Memory hardware

All memory chips, which are used to store individual bits of data, work on the same basic principle. Each bit of data has either of two values, 0 or 1. The presence of a small electrical charge indicates that the bit is *on* and therefore has a value of 1. If a bit has no charge, it has a value of 0. You may recall that 8 bits comprise a *byte,* and it takes one byte to store a single character.

Over the past ten years, the demand for memory soared. Technology reacted to meet that demand with a great variety of different memory chips. The following is a listing of the chips that dominate the market.

## The DRAM chip

The most common memory chip in PC, XT, and AT computers is the *dynamic random-access memory* (DRAM) chip. The DRAM chip has sixteen metal pins that protrude downward from the bottom of the chip, as shown in Figure 5-1. To install a DRAM chip, you push these pins into sockets or solder them onto the motherboard or memory add-on board.

The memory storage capacity of DRAM chips can range anywhere from 16K bits to 4,096K bits and beyond. The 16K bit chips were placed in the original IBM PCs, but today, the most commonly used size is 1,024K bits. (Note that the memory sizes are stated in bits and not bytes. Each DRAM chip stores one bit of a byte. When the CPU requests data from memory, it accesses eight chips simultaneously; each chip provides one-eighth of the data.)

DRAM memory is arranged in banks of nine chips. Eight of the chips store a single bit of each byte, and the ninth bit stores a cross-check or *parity value.* This parity value ensures that the chips are functioning correctly. (For a more detailed explanation of this setup, see the sidebar titled "How Parity Checks Work.")

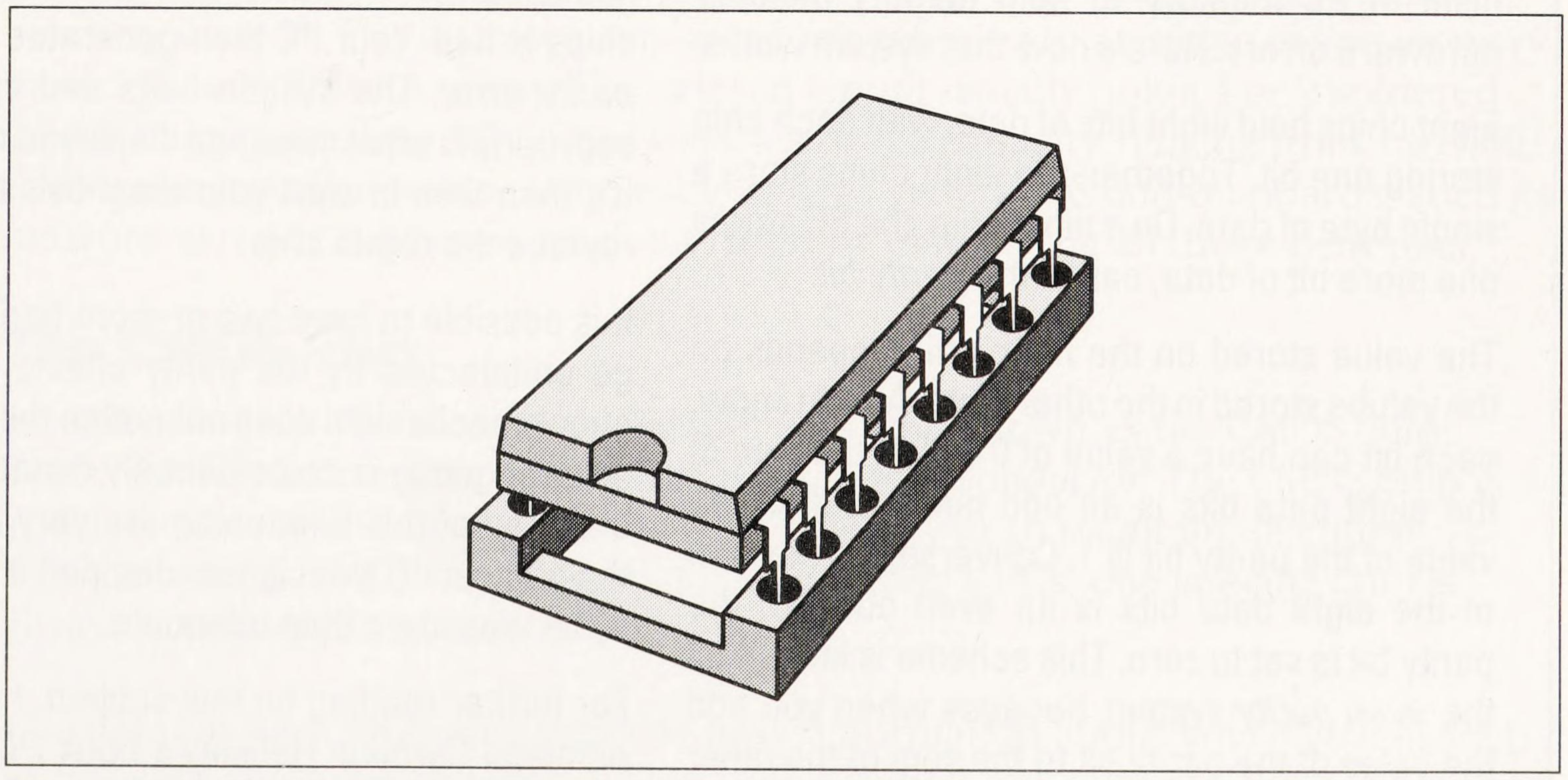


**Figure 5-1:** A DRAM chip.

Like all PC memory chips, a DRAM chip stores data as an electrical charge of about five volts. The charge in a DRAM chip fades over time, and the chip loses its memory — hence the name, *dynamic* RAM. For the DRAM chip to maintain accurate data, the charge must be refreshed every few thousandths of a second. This can lead to delays: Before the CPU can save data on the chip, it may be forced to wait until the chip's charge is refreshed.

### The SRAM chip

Dynamic RAM chips do not require much power, but their charges must be refreshed frequently. The *static* RAM chip, or SRAM, requires more power but doesn't need to be refreshed. SRAM chips are somewhat power-hungry and expensive to produce, but they are very fast. The CPU doesn't have to wait for charges to be refreshed.

In the last chapter, you learned that disk caches can significantly increase overall system performance because memory access is much faster than disk access. Well, as you now know, some types of memory are faster than others — SRAM chips are a lot faster than DRAM chips. In high-performance systems, SRAM chips often are used as a memory cache. They cache memory data in much the same way that a disk cache stores disk data. The SRAM cache is used

TECHNICAL STUFF

### How Parity Checks Work

Nobody wants a computer to make mistakes. PC designers added a simple cross-check mechanism to PC memory to help identify memory hardware errors. Here's how that system works.

Eight chips hold eight bits of data, with each chip storing one bit. Together, the eight chips store a single byte of data. On a ninth chip, the PC stores one more bit of data, called the *parity bit.*

The value stored on the ninth chip depends on the values stored in the other eight. As you know, each bit can have a value of 0 or 1. If the sum of the eight data bits is an odd number, then the value of the parity bit is 1. Conversely, if the sum of the eight data bits is an even number, the parity bit is set to zero. This scheme is known as the *even parity system,* because when you add the value of the parity bit to the sum of the other eight bits, the result is always an even number.

If the system finds that the sum of the nine bits is an odd number, it knows that a hardware failure has occurred — that is, that one of the memory chips is bad. Your PC then generates a *memory parity error.* The system halts and you see an appropriate error message displayed on-screen. It's then time to start your diagnosis to find and replace the rogue chip.

It is possible to have two or more bad chips that go undetected by the parity check. The parity check mechanism does not notice the problem if an even parity is coincidentally maintained. The chances of this happening are very remote, so the original PC's designers decided a single parity bit was more than adequate.

For further reading on this subject, I highly recommend Caroline Halliday's book *PC SECRETS,* published by IDG Books Worldwide.

to store frequently or recently accessed memory data. When the CPU requests some data, the SRAM cache is checked first. If the data is available in the cache, the SRAM chips pass the information to the CPU very quickly.

NOTE

Systems that use SRAM chips for a cache usually are referred to as having *cache memory.* Such systems consistently outperform similar systems that don't have cache memory. Although systems with cache memory are a few hundred dollars more expensive, the performance gains are significant. (In fact, the performance improvements from even a small memory are so significant that Intel built an onboard 8K memory cache directly into the 486 chip.) The bigger the cache, the better. Cache memory added to the system board between memory and CPU is usually 64K to 256K in size.

To date, no PCs provide a way to add cache memory after the fact. The cache memory must be installed during manufacturing. If you want a high-performance system, get one that uses cache memory because you won't be able to install it later.

## SIMMs

DRAM chips were ideal in the early PC days, when memory was an expensive luxury and users only added small amounts of it — 256K on this machine and 384K on that one. Nowadays, users add memory in chunks of 4MB or more. That's a lot of chips and circuit-board real estate. And as you can imagine, it's easy to bend the pins on DRAM chips during installation — I know I've bent my share.

As high-memory systems became more popular, manufacturers began shipping DRAM memory chips already installed on small circuit boards. The motherboards had special memory sockets designed to receive these small memory boards. These chips-on-a-board are referred to as *single in-line memory modules,* or SIMMs for short. The printed circuit usually holds 4 or 9 soldered DRAM chips (see Figure 5-2) that have a storage capacity ranging from 1 to 8MB. SIMMs are installed in much the same way as general PC add-on boards, such as network cards and internal modems, thereby eliminating all those bent pins.

## The CMOS chip

All 286 or better PCs have a special memory chip known as the CMOS chip. CMOS stands for *complementary metal-oxide semiconductor.* The CMOS chip is a special memory chip that requires very little power to maintain its charge. Although several CMOS chips are installed in today's PCs, one specific chip is always considered to be *the* CMOS chip.

The data on the CMOS chip records basic information about your system, such as the date and time, the floppy disk configuration, the number and type of hard disks installed, etc. If you install some new equipment or modify your system configuration, you can update this information from the main SETUP program.

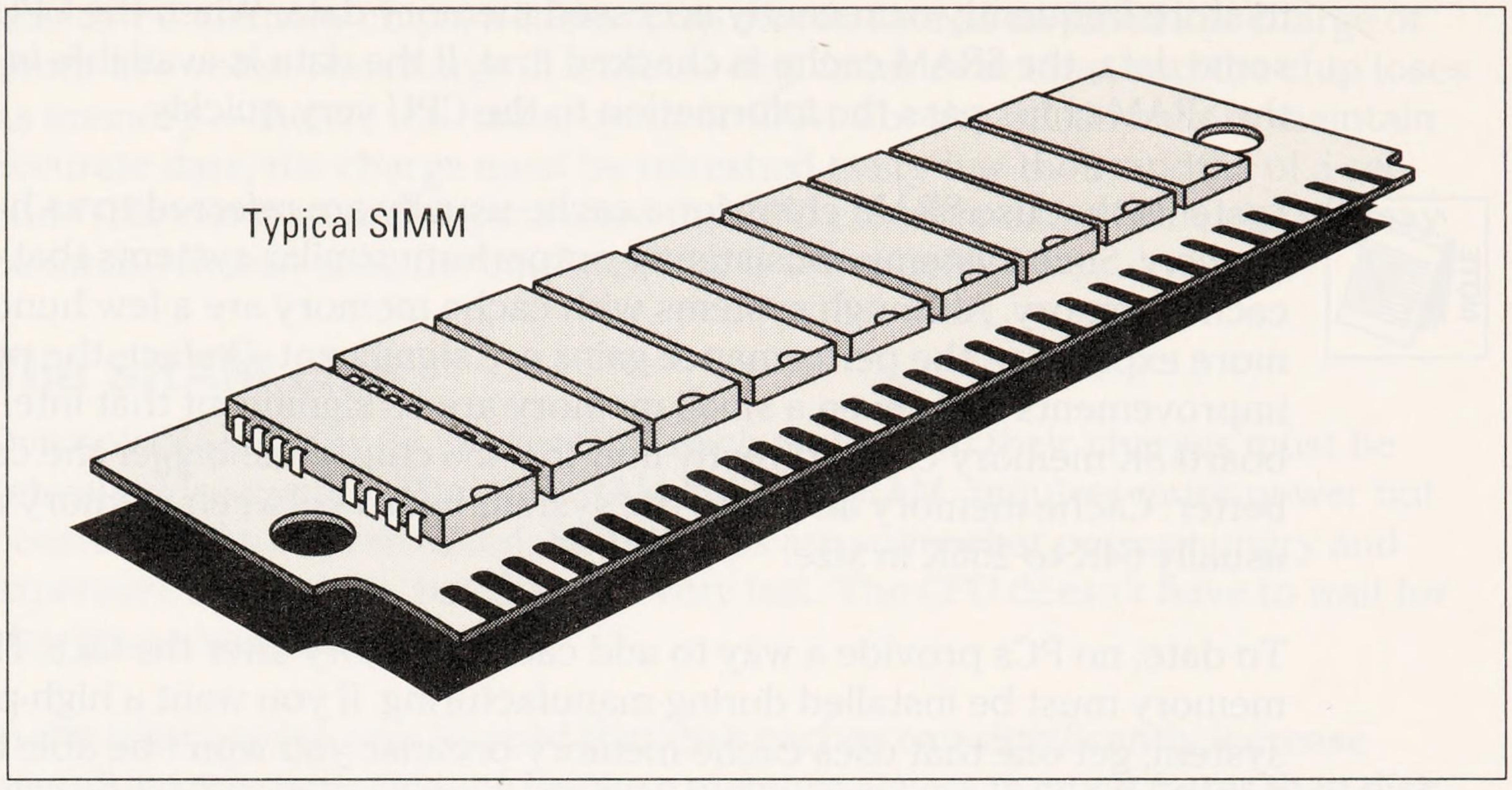


**Figure 5-2:** A SIMM board.

A small battery inside the PC chassis supplies power to the CMOS chip. (Often, the battery is attached with a Velcro strip.) That's how the computer remembers the date and time even though you turn off the power supply. The battery can last as long as two years.

## Memory speed issues

Some types of memory are fast, and some are not so fast. The speed of a memory chip is rated in *nanoseconds* (NS for short). One nanosecond equals one-billionth of a second.

The general speed rating is referred to as the chip's *access time.* The smaller the rating, the faster the chip. PC memory access times can range anywhere from 140NS, at the very slow end of the spectrum, to 60NS or better at the other.

It's not always sensible to install the fastest memory available. As my grandfather would say, there is little use in hiring a faster mortar-mixer if you've got a slow bricklayer. The same goes for memory chips. You're wasting your money to buy memory chips that are too fast for the speed of the CPU; the chips will simply sit idle longer. Look in your computer handbook or consult with the PC manufacturer before buying memory chips to upgrade your system.

The CPU speed, referred to as *clock speed,* is rated in *megahertz* (MHz). The original IBM PC AT was rated at 6 MHz; now, it is not impossible to see 66 MHz PCs in the home. The clock speed controls the tempo at which the CPU performs operations. The faster the clock speed, the more quickly the CPU completes tasks. Although we have witnessed a huge increase in PC clock speed, we have not seen the same improvements in memory-chip speed.

Sometimes, when the CPU needs to get data from memory or put data into memory, it must wait for the memory chips to respond. The memory may be just plain slow, or the refresh cycle may be to blame. (Remember, while a DRAM chip is being refreshed, it cannot save data.) The average number of *clock cycles* (or *ticks*) the CPU has to wait while the memory does its share is called the *wait state.* The optimum PC has zero wait states, which indicates that the CPU is never delayed waiting for the memory chips to respond.

After data is accessed from a specific memory location, the chip must prepare itself for the next access. The delay incurred while the chip prepares for concurrent access is referred to as the *precharge time.*

A few years ago, PC designers faced a dilemma: Even the fastest DRAM chips couldn't keep pace with the fastest CPUs, and SRAM chips were too expensive and too power-hungry. Rather than make a faster chip, manufacturers solved the problem by redesigning the memory storage architecture to improve performance. The concept of memory *interleaving* was introduced.

Interleaved memory is organized on the principle that memory is most often accessed in contiguous blocks. For example, the CPU requests data from memory location 10000 and then immediately accesses the memory at location 10001. To get around precharge time delays, the memory is divided into two major parts: even memory locations and odd memory locations. With interleaved memory, the system accesses one chip to get data from address 10000 and another chip to get data from address 10001, thereby avoiding the precharge delay.

It appears that systems engineers have exhausted the configuration tricks that can be used to squeeze the most out of DRAM-based memory chips. Power-hungry users, myself included, eagerly anticipate the introduction of the successor to DRAM chips. Only with a fundamental change in technology will memory chips be able to keep pace with the improvements in CPU performance.

## Memory addressing

The process of optimizing memory involves telling the memory manager to include or exclude specific memory locations when you load certain programs. Every memory location has an address, just like every house on a street has an address. To master the memory manager, you need to understand the PC memory addressing scheme. And to understand that scheme, you must first understand hexadecimal numbers.

### Hexadecimal numbers

Most people are familiar with the decimal, or base 10, numbering scheme, in which any single digit can represent ten different values, zero through nine. If we had been created with eight fingers on each hand, instead of five, we might have adopted a 16-based numbering scheme.

The memory addressing scheme in PCs uses a 16-based numbering system known as *hexadecimal* or *hex.* In this numbering scheme, each digit can represent one of sixteen unique values.

In Latin-based languages, we use special characters to denote numbers: a circle represents a zero (0), a vertical line represents a one (1), a curly line with a flat base represents a two (2), and so forth. The same is true for hexadecimal numbers, but there is a problem for the six numbers greater than nine. What characters do we use to represent them? Trust mathematicians to come up with a boring solution.

In the case of hexadecimal numbers, the letters A through F are used to represent the numbers 10 through 15. So the sixteen numbers that represent the base hexadecimal numbers are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F.

In the decimal system, we represent numbers larger than nine by using digits to represent the tens, hundreds, thousands, etc. For example, the number 625 means 6 hundreds, 2 tens, and 5 singles. Most people find this scheme intuitive and never really think about it. The same scheme applies to hexadecimal numbers. In decimal, the numbers increase in tens; in hexadecimal, the numbers increase in units of sixteen — 16s, 256s, 4096s, etc.

The hexadecimal number 235 means two 256s, three 16s, and 5 singles. That translates to the decimal number 565. The following list shows some hexadecimal numbers and their decimal equivalents:

| ***Decimal*** | ***Hexadecimal*** |
|---|---|
| 3 | 3 |
| 9 | 9 |
| 10 | A |
| 13 | D |
| 17 | 11 |
| 55 | 37 |
| 256 | 100 |
| 266 | 10A |

As you can see, it is difficult, if not impossible, to distinguish hexadecimal numbers from decimal ones when no letters are used. There are, however, a few popular conventions for identifying hexadecimal numbers. One convention is to place the characters *0x* or *$* before the number. Alternatively, the lowercase letter *h* is used as a suffix to the number. For example, the hexadecimal equivalent of the decimal 55 can be written as *0x37, $37,* or *37h.*

Most programmers and systems engineers are conversant in hexadecimal arithmetic. I know some people who can add up their grocery bill in hex. Fortunately, you really only need to understand the numbering scheme in principle in order to manage PC memory. By the way, I'm 24 years old in hexadecimal!

## Segments and offsets

The original IBM PC used Intel's 8088 microprocessor (or CPU). Shortly after the IBM PC was released, the 8086 microprocessor gained popularity. It was installed in some IBM PC compatibles as well as in the IBM PS/2 model 25 and model 30.

The 8088 uses an 8-bit data bus, whereas the 8086 uses a 16-bit data bus (more about buses later). In all other respects, the 8086 and 8088 chips are identical. They form the foundation of the entire PC family of CPUs, and all newer Intel microprocessors, including the 80486, can emulate and operate like fast 8086s.

The 8086 chips use a set of *registers* to manage and manipulate data. For example, suppose that you want to multiply two whole numbers. Each number is moved into a different register, and the CPU is instructed to multiply them together. Each register is 16 bits in size.

The registers are also used to access data stored in memory, but the largest value a 16-bit number can have is 64K, so a single register can only access 64K worth of single-byte memory addresses. This value is actually 2 to the power of 16, or 65,536 to be accurate. Even the original PC engineers recognized that a PC with 64K of memory wouldn't be very functional, so they decided to use two registers to store memory addresses. They devised a system that allows 1,048,576 bytes (one megabyte) of memory to be accessed. Surely no program would ever need to address more memory than that!

Here's how the system works. Every byte in memory has a unique address, just like every office in a building has a unique address. A PC memory address comprises two parts, the *segment* and the *offset.* These are similar to the floor and office number of an office building.

The 1MB of memory is subdivided into major blocks called *segments.* Each segment starts 16 bytes beyond the previous one. To identify a specific memory location, you first identify the segment. This is like specifying the floor on which a particular office is located. The *offset* specifies how many bytes there are between the beginning of the segment and the precise memory location. This is like specifying the room number of the office.

To specify a memory address, you must identify both the segment and the offset. Normally, you indicate the segment and offset as follows: *segment:offset.* A colon (:) separates the segment and offset values. The CPU stores the segment in one 16-bit register and the offset in another.

Values are specified in hexadecimal, and the numbers are usually zero-padded to four digits. For example, the memory address at segment B800, offset 0 is written as *B800:0000,* and the memory address at segment 40, offset 17 is written as *0040:0017.* (Although the numbers are hexadecimal, the 0x, h, or $ symbols are often not added because memory addresses are always specified in hexadecimal.)

Some memory management tasks are performed on a range of memory rather than on a single location. If you see two four-digit numbers separated by a minus (–) sign, the address actually identifies a memory range. The value before the minus sign indicates the first segment in the range, and the second value indicates the last segment in the range, as follows: *StartSegment-EndSegment.*

## Multiple Memory Addresses

In the *segment:offset* scheme, one specific memory location can have multiple addresses. In other words, many different addresses can point to the same memory location.

There is a new segment address every 16 bytes: Segment 0000 starts at the beginning of addressable memory; segment 0001 starts at the 16th byte; segment 0002 starts at the 32nd byte, and so on. The memory offset value can range from 0 to FFFF (65,536 in decimal). So the address 0002:0004 is the same as 0001:0014 because they both point to an address 14 bytes (20 decimal — 16 + 4) from the beginning of addressable memory.

Imagine an office tower that has 16 offices per floor. The 7th office on the 20th floor would normally have the address *Floor 20, Room 7.* If the building used a scheme similar to PC memory addressing, the office could also have the address *Floor 19, Room 23,* or even *Floor 18, Room 39.* Wild.

Behind the scenes, the actual memory address is linear. The first byte of memory has address 0, and the last byte has address 1,048,575. When the CPU needs to access a specific byte of data, the segment:offset address is converted to a linear address. This is achieved by multiplying the segment by 16 and adding the offset, which yields a 20-bit number identifying the precise linear memory location.

For example, the memory address 1000:2345 can be converted to a linear address by multiplying 1000h by 16. Sixteen in hexadecimal is 10h, so the result is 10000h. The offset of 2345h is added to 10000h, resulting in a linear address of 12345h.

Using the office analogy again, you can convert the address of the 7th room on the 20th floor to a linear room number by multiplying the floor number (20) by 16, because there are 16 rooms on each floor, and then adding 7. The resulting linear room number is 327. It's a good thing this scheme isn't used in airport parking lots; I'd never find my car.

If you don't intend to do any programming, you may be wondering why you need to know how the CPU addresses memory. The answer is that you may encounter these memory addresses when you install device drivers for specialty hardware or add commands to your CONFIG.SYS file to fine-tune the memory manager. For example, the following line is extracted from my CONFIG.SYS file:

```
device=c:\dos\emm386.exe ram win=dd00-dfff win=da00-dcff
```

The WIN statements at the end of the line instruct the memory manager EMM386 to reserve the specified area of memory for use by Windows. Later in the chapter, you learn much more about EMM386 and its options.

Memory addresses using the segment and offset notation can range from 0000:0000 to FFFF:FFFF. (Remember, addresses are specified in hexadecimal, and the highest hexadecimal number is F.)

## Major types of memory

The 8086 family of microprocessors uses a 20-bit addressing scheme that yields a maximum addressable memory of 1MB. When Intel introduced the 80286 (or 286 for short), they added new memory capabilities, but they also made sure that the 286 could function just like a fast 8086 chip. There would have been a riot if software that used to run on the 8086 would not run on the 80286. Some new applications were written that utilized the memory scheme introduced in the 286, but users with 8086 systems could not run these applications. You may be aware of software that will only run on 286s or better — for example, Lotus 1-2-3 Release 3.

Even with the added memory capabilities of the 286, users still wanted more. The 386 family of microprocessors was introduced, sporting yet more advanced memory management features. Once again, the chip had to be backward-compatible — this time, with the 286 and 8086 microprocessors. Over time, software was written to take advantage of the 386 memory management capabilities. Windows 3.1 (in Enhanced Mode), Paradox 386, and even some features of DOS are examples of such programs.

Memory comes in a variety of flavors, and some applications require memory that is configured a certain way. In this section, you learn about the various memory architectures available on each CPU platform. The model of a memory warehouse is used to help clarify the various options.

### How to check your system

How much you can improve your system depends on how much memory you have installed and the type of CPU in your system. If you are not sure about your hardware base or your current memory setup, you can use the DOS 6 program MSD to find out.

If you use an earlier version of DOS, you may have the MSD program and not know it — it was included in the Windows 3.1 package. (There were even fledgling versions of MSD shipped with Microsoft Word 5.5.) If you have DOS 6 (or DOS 5), you can search for MSD on your hard disk by entering the following command:

```
c:\> dir \msd.exe /s
```

Alternatively, you can use the QFIND program on Bob's Shareware Collection disks to search for MSD.EXE.

To run MSD, simply enter the MSD command without any options, as follows:

```
c:\> msd
```

After MSD snoops around your system, it prompts you to press a key and then presents you with the main menu.

To find out your processor type, select the Computer option by clicking on it with the mouse or by pressing P and looking at the Processor entry. To review your memory configuration, select the Memory option in a similar manner. Figure 5-3 illustrates sample displays of the information generated by MSD for the Computer and Memory options.

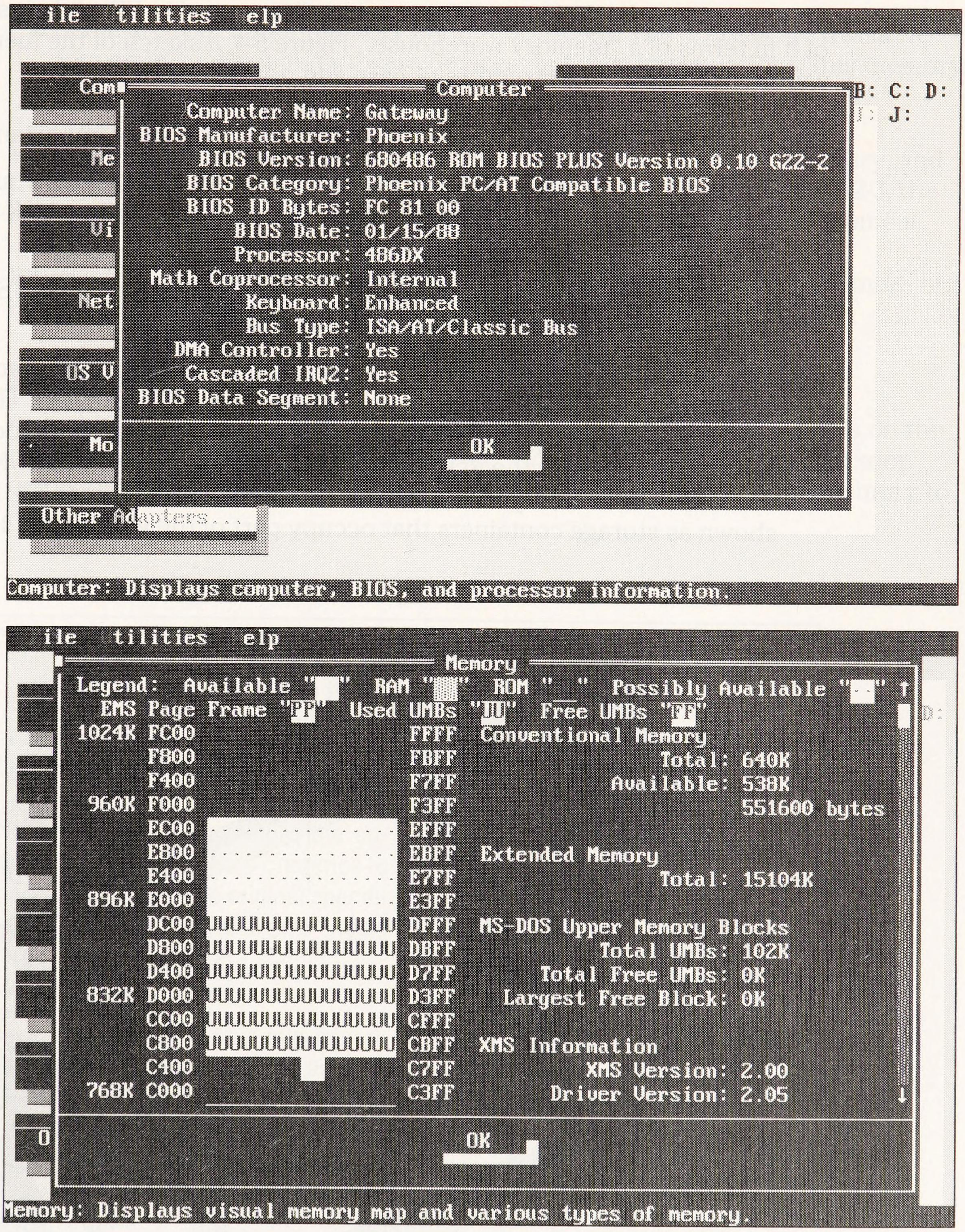
## Base and upper memory

Most DOS users have heard about the 640K barrier or ceiling. This ceiling represents the most memory a program can use, and it continues to be a source of frustration to this day. But wait a minute — in the last section, you learned how the original PC engineers devised a memory addressing system that allowed the CPU to access 1MB of memory. So why is there a 640K ceiling rather than a 1MB ceiling?

Although there is 1MB of addressable memory in a PC, not all of it is directly available to DOS programs. The area between 640K and 1MB is reserved for use by special adapter boards and ROM-BIOS. Some memory, for example, is reserved for the video display. This reserved area of memory from 640K to 1MB is known as *upper memory*.

It is safe to say that the amount of memory address space dedicated to upper memory is very generous, especially when you consider how precious base memory is. In many cases, one third or more of upper memory is not used at all. If you have a 386 or later system, you can steal this area of memory and use it for storing DOS programs. You learn how to do so later in this chapter.

Over the past few years, there have been many changes in memory technology. Unfortunately, there have been just as many changes in memory terminology. Some memory terms used in the late 1980s now have a different meaning. For example, in old literature, you may see upper memory referred to as *high*



```
Computer
     Computer Name: Gateway
 BIOS Manufacturer: Phoenix
      BIOS Version: 680486 ROM BIOS PLUS Version 0.10 G22-2
     BIOS Category: Phoenix PC/AT Compatible BIOS
     BIOS ID Bytes: FC 81 00
         BIOS Date: 01/15/88
         Processor: 486DX
  Math Coprocessor: Internal
          Keyboard: Enhanced
          Bus Type: ISA/AT/Classic Bus
    DMA Controller: Yes
     Cascaded IRQ2: Yes
 BIOS Data Segment: None

Computer: Displays computer, BIOS, and processor information.
```

```
Memory
Legend:  Available "  "  RAM "▒▒"  ROM "  "  Possibly Available ".."
  EMS Page Frame "PP"  Used UMBs "UU"  Free UMBs "FF"
1024K FC00                  FFFF  Conventional Memory
      F800                  FBFF                  Total: 640K
      F400                  F7FF              Available: 538K
 960K F000                  F3FF                         551600 bytes
      EC00 ................ EFFF
      E800 ................ EBFF  Extended Memory
      E400 ................ E7FF                  Total: 15104K
 896K E000 ................ E3FF
      DC00 UUUUUUUUUUUUUUUU DFFF  MS-DOS Upper Memory Blocks
      D800 UUUUUUUUUUUUUUUU DBFF             Total UMBs: 102K
      D400 UUUUUUUUUUUUUUUU D7FF        Total Free UMBs: 0K
 832K D000 UUUUUUUUUUUUUUUU D3FF      Largest Free Block: 0K
      CC00 UUUUUUUUUUUUUUUU CFFF
      C800 UUUUUUUUUUUUUUUU CBFF  XMS Information
      C400                  C7FF            XMS Version: 2.00
 768K C000                  C3FF         Driver Version: 2.05

Memory: Displays visual memory map and various types of memory.
```

**Figure 5-3:** Computer and memory displays from MSD.

*memory*. Worse still, the term high memory now has a different meaning. This can be a real source of confusion. The good news is that it appears terminology has finally matured, so you should always use the terms as I describe them in this chapter.

To understand the memory management scheme used in PCs, it helps to think of it in terms of a "memory warehouse." Figure 5-4, a sketch of the memory warehouse, shows the structure of base and upper memory.

The memory warehouse in Figure 5-4 illustrates a number of important details:

- The DOS command interpreter, COMMAND.COM, automatically uses the first portion of the 640K base memory. This leaves less than 640K for your programs.
- Device drivers (such as mouse and network drivers) and other TSR programs use up still more of your base memory.
- Although 8086 computers can address 1,024K of memory, most computers do not actually have that much general-purpose RAM installed. The memory addresses in upper memory are reserved for memory located elsewhere. For example, the ROM-BIOS memory is located on the ROM chips, and the video memory is located on the video card. In Figure 5-4, they are shown as storage containers that occupy part of the warehouse space.

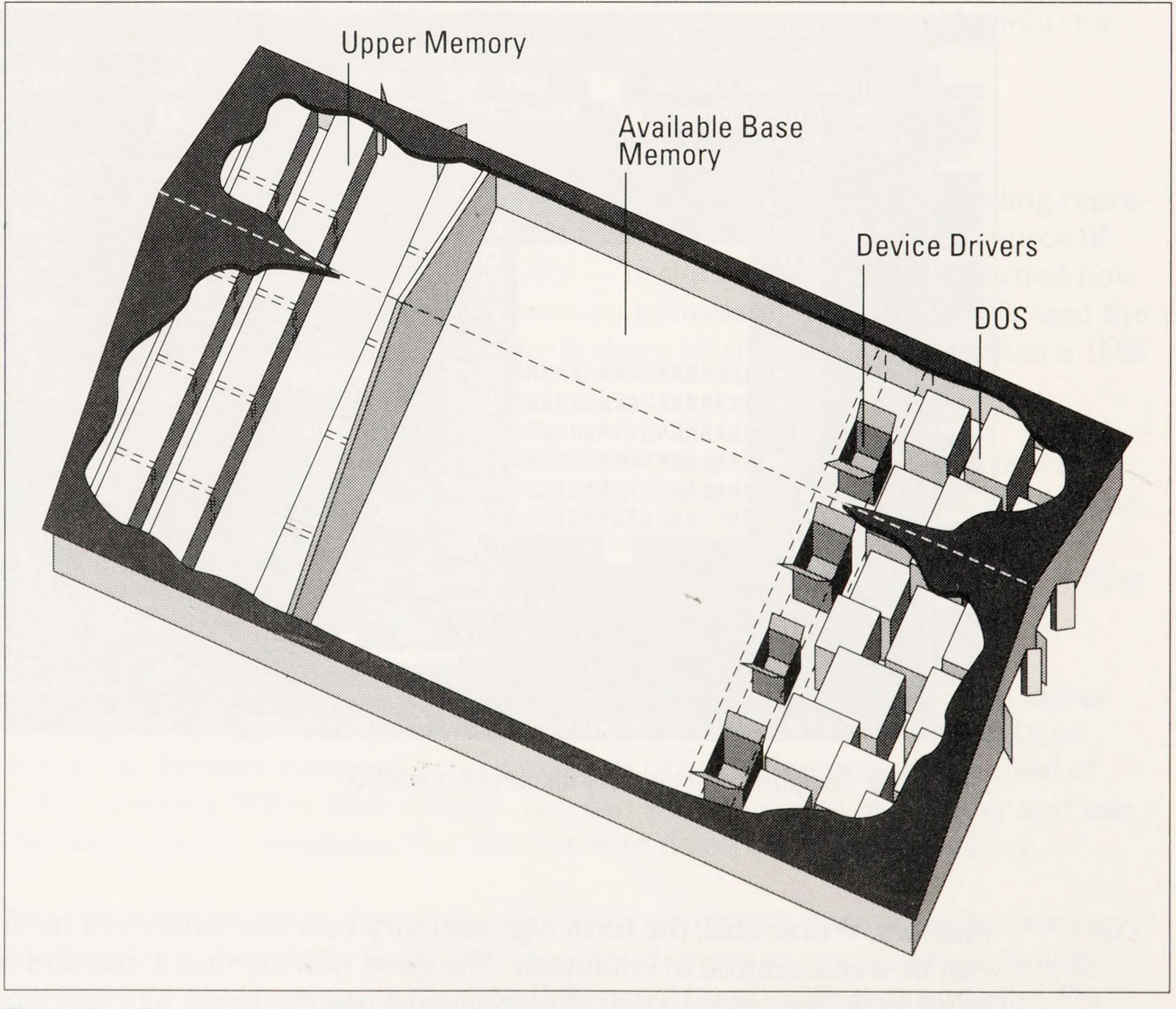


**Figure 5-4:** The standard memory warehouse.

- Some areas of upper memory are not occupied. These areas are available for use by additional hardware devices, but in many situations, this memory is wasted and not used.

When a program is executed, it is loaded into the bottom of base memory, and the remaining base memory is available for data. In the case of Lotus 1-2-3, the data is the spreadsheet, and in WordPerfect, the data is a letter or document.

When you load a large program like Lotus 1-2-3, it may occupy 200K or more of memory, and the size of data area often is reduced to 300K or less.

## Expanded memory

Arguably the most successful PC program and the one that first put PCs on the desktops of corporate America was Lotus 1-2-3. But corporate America soon demanded bigger spreadsheets, and PCs simply didn't have enough memory to do the job.

### The History of EMS

The original expanded memory specification was developed by Lotus and Intel and called LI-EMS. Believe it or not, this first specification was dubbed EMS version 3.0. Presumably, the authors believed we would give more credence to a third-generation product. The original specification detailed how memory hardware should be engineered and how software should communicate with it.

Microsoft joined forces with Lotus and Intel to produce a revised specification known as LIM-EMS version 3.2. The primary change in the specification between 3.0 and 3.2 was the removal of strict hardware controls, which in turn allowed more vendors to produce expanded memory products.

A number of other PC vendors believed that LIM-EMS 3.2 was too conservative and didn't utilize the full potential of expanded memory. Ashton-Tate (now part of Borland International), Quarterdeck Office Systems (the DESQview folks), and board-maker AST Research banded together and developed a better mousetrap, called the *Enhanced Expanded Memory Specification,* known as EEMS. (Aren't you glad they didn't call it the ATQOSASTR-EMS?)

EEMS supported all the features of LIM-EMS 3.2 and then some. The most important new feature was the capability to load and run programs from the EMS page frame. The original specification was really oriented to managing data and not programs in expanded memory. By adding support for programs, EEMS provided an excellent vehicle for *multitasking* programs — that is, running more than one program at a time and switching between them.

While all this was taking place, other vendors, such as IBM, were developing their own EMS enhancements. The final chapter in the EMS story was written by the LIM team. In a cooperative effort to settle on one standard, the LIM team took the best features of all the hybrid EMSs and introduced LIM-EMS version 4.0. This specification incorporated most of the features of EEMS, EMS 3.2, and other vendors' variations. Today, LIM-EMS 4.0 stands as the dominant expanded memory specification for the PC industry.

Three major companies in the PC business got together to devise a way to make more memory available: Lotus Development Corporation was the spreadsheet company, Microsoft Corporation was the operating system company, and Intel Corporation was the hardware manufacturer. They called their solution *expanded memory*.

As Figure 5-4 illustrates, there is a lot of wasted space in upper memory. The obvious solution to the memory shortage problem was to make better use of upper memory. But upper memory could only offer a few hundred thousand extra bytes at most, and it was becoming clear that users wanted spreadsheets that were two or three megabytes in size. The answer to this dilemma was *memory paging*.

Lotus, Intel, and Microsoft designed the *expanded memory specification,* or EMS. This specification often is called the *LIM-EMS,* named after the three main contributors (Lotus, Intel, and Microsoft). The theory behind LIM-EMS is quite simple. A 64K area of upper memory is reserved for use as expanded memory. This area, called the *EMS page frame,* is subdivided into four 16K pieces. Those four pieces can be taken from a pool of 32MB of memory.

A program cannot access all the expanded memory at one time. Rather, it must manage the data in 16K pieces. The process of assigning different blocks of memory into the page frame is known as *bank switching*.

In the memory warehouse sketch shown in Figure 5-5, the expanded memory pool is represented by 16K container trucks. This memory is separate from the main 1MB warehouse and must be shunted in small pieces into the warehouse before DOS can access it.

The truck analogy isn't quite accurate because the 16K pieces of expanded memory aren't actually moved into upper memory. A memory manager fools DOS into thinking the memory is located in upper memory. When DOS tries to manipulate the data in the expanded memory area, the memory manager intercepts the task and makes the real changes directly to the real memory, which is located somewhere in the 32MB expanded memory area.

The expanded memory solution for 8086 computers is, at best, a work-around. Although it provides a lot more memory to applications, the bank switching process is not efficient. Dealing with memory in small 16K pieces is restrictive and time-consuming. Ideally, software needs to access huge blocks of contiguous memory.

## Extended memory

When Intel developed the 80286 microprocessor (286), the company decided to implement a new memory architecture that provided access to large volumes

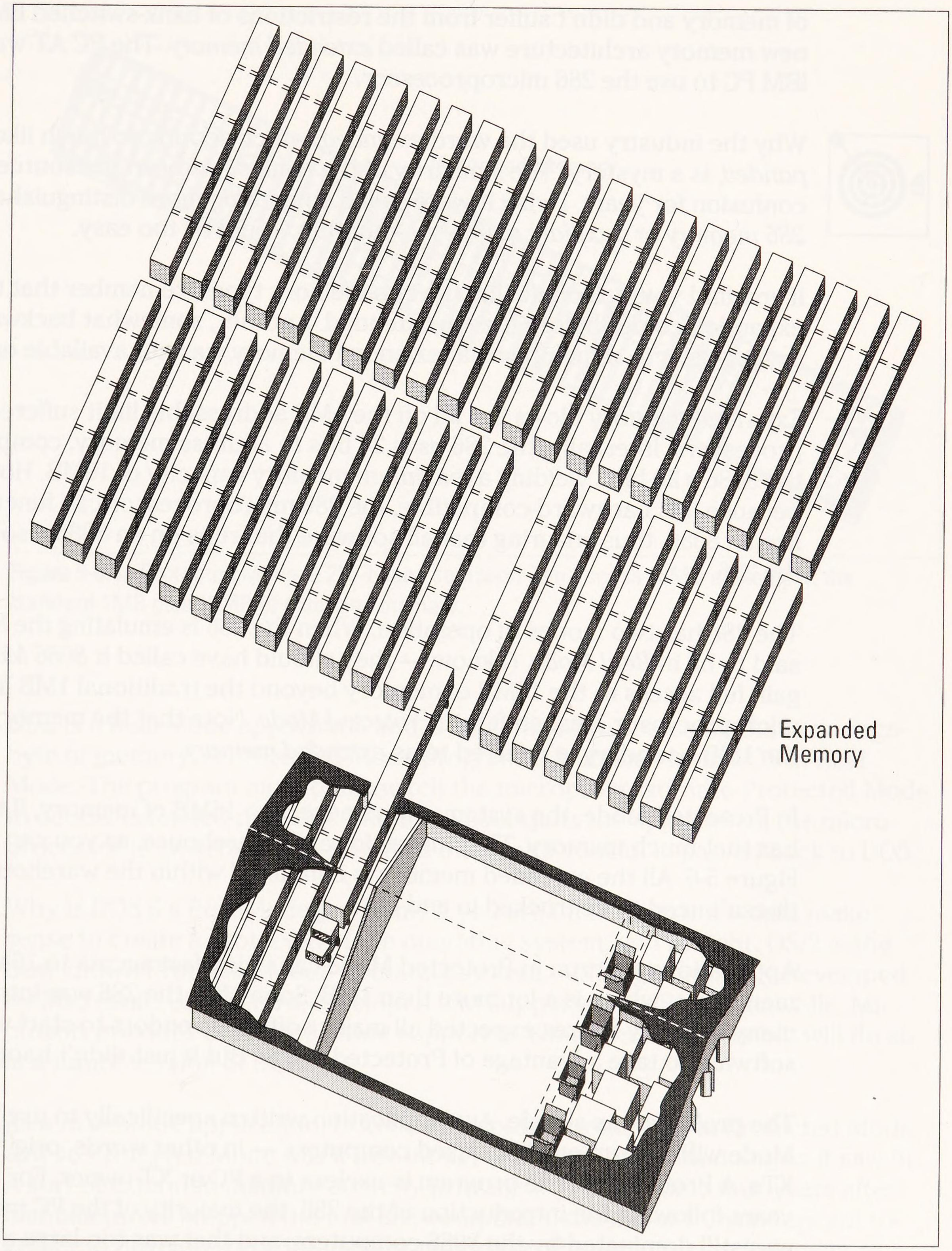

Expanded
Memory

**Figure 5-5**: Expanded memory is divided into 16K pieces and accessed from a 64K area of upper memory.

of memory and didn't suffer from the restrictions of bank-switched EMS. The new memory architecture was called *extended memory.* The PC AT was the first IBM PC to use the 286 microprocessor.

Why the industry used the word *extended,* which sounds so much like *expanded,* is a mystery. The similarity of the names has been the source of confusion for years. I wish it were called something more distinguishable, like *286 memory* or *auxiliary memory* — but that would be too easy.

If you find yourself confusing the two memory types, remember that the *p* in expanded comes before the *t* in extended. Another, somewhat backward, memory jogger is to remember that e*xt*ended memory was not available on the *XT.*

Extended memory blows the lid off the 1MB addressable limit suffered by 8086 processors. Internally, the 286 uses 24 bits to address memory, compared with the 8086's 20 bits, yielding a maximum memory capacity of 16MB. However, because it is backward-compatible, the 286 microprocessor can function just like an 8086, thus ensuring that all software that runs on an 8086 also runs on the 286.

The 286 has two modes of operation. When the 286 is emulating the 8086, it is said to be in *Real Mode.* (I know — they should have called it *8086 Mode.*) To gain full access to the 15MB of memory beyond the traditional 1MB limit, the microprocessor must shift into *Protected Mode.* Note that the memory above the 1MB boundary is referred to as *extended memory.*

In Protected Mode, the system enjoys access to 16MB of memory, if the PC has that much memory. That makes for a big warehouse, as you can see in Figure 5-6. All the extended memory is contained within the warehouse and doesn't need to be trucked in and out.

Applications that run in Protected Mode can enjoy fast access to 16MB of memory — which is a lot more than 1MB. Soon after the 286 was introduced, many industry figures expected all major software vendors to start writing software to take advantage of Protected Mode. But it just didn't happen.

The problem was simple. Any application written specifically to use Protected Mode will not run on 8086-based computers — in other words, original PCs and XTs. A Protected-Mode program is useless to a PC or XT owner. For the first few years following the introduction of the 286, the majority of the PC marketplace was still dominated by the 8086 computers, and that was too large a market to ignore. Nonetheless, a few companies did introduce products written exclusively for Protected Mode. Lotus 1-2-3 Release 3 was the most popular. Even though Lotus Development introduced Release 3, the company made it clear that it would continue to enhance the Release 2 version of the product to keep 8086 customers happy.

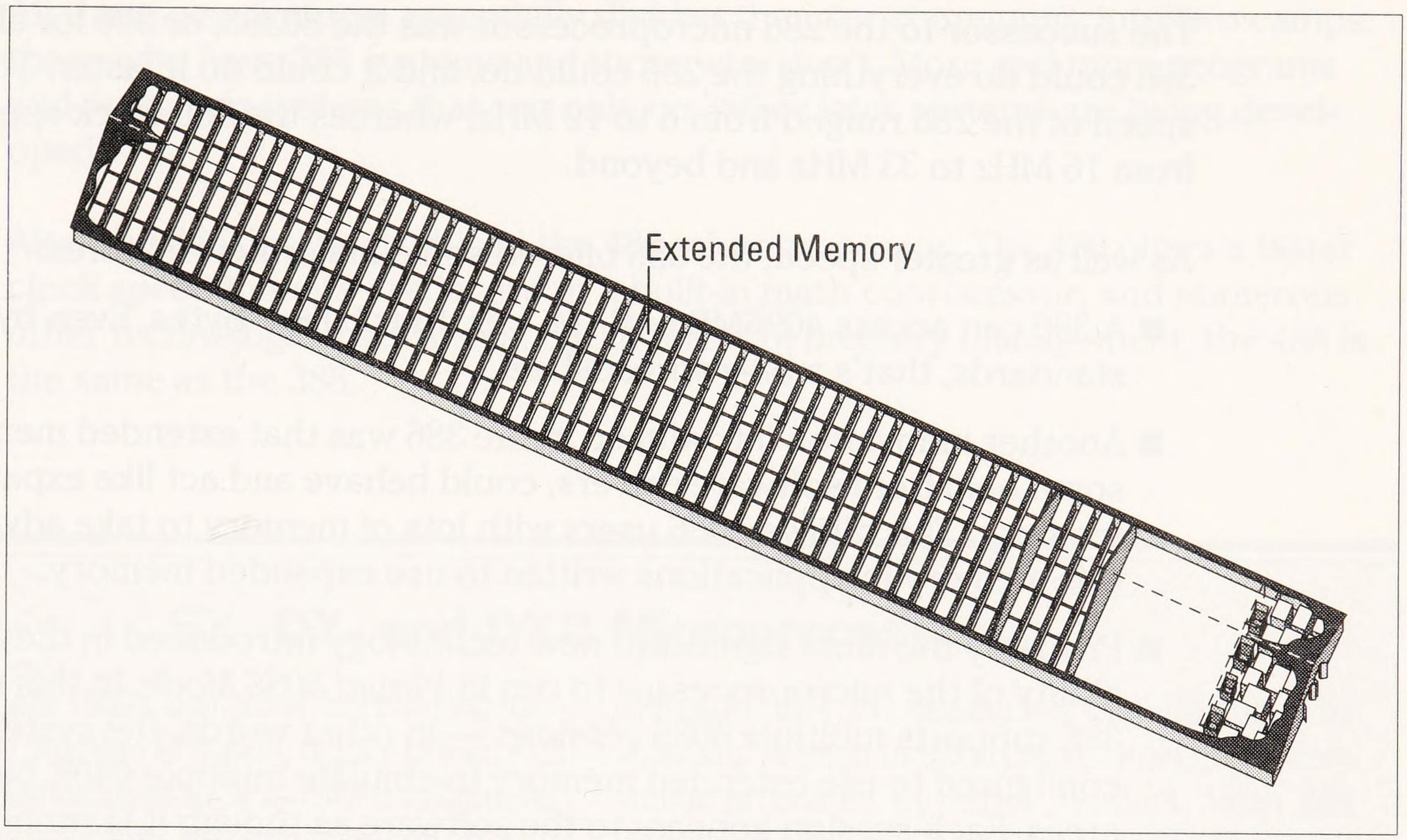


**Figure 5-6:** In Protected Mode, a 286-based computer can access 16MB of memory: the standard 1MB plus 15MB of extended memory.

DOS is a Real-Mode application and is still limited to accessing the first megabyte of memory. A Protected-Mode program is launched from DOS in Real Mode. The program must then switch the microprocessor into Protected Mode to access extended memory. When the user quits the application, the microprocessor is switched back into Real Mode and control is passed back to DOS.

Why is DOS 6 a Real-Mode program? If you are thinking that it would make sense to create a Protected-Mode operating system, you're right. OS/2 is the best known Protected-Mode operating system. Although originally developed by Microsoft, OS/2 is now developed and supported by IBM. Meanwhile, Microsoft provides Protected-Mode support in Windows and probably will do so in a future version of DOS.

As a Real-Mode application, DOS had no control or interest in Protected Mode, and so Protected Mode was a free-for-all, with each program doing as it saw fit. A sort of extended-memory anarchy prevailed. In 1988, some four years after manufacturers shipped the first 286 computers, the major PC vendors got together to develop some industry standards on the use of extended memory. The folks that were most involved in expanded memory (Lotus, Intel, Microsoft, and AST) released a new specification for extended memory. Not surprisingly, this specification was dubbed the *extended memory specification,* or XMS.

The successor to the 286 microprocessor was the 80386, or 386 for short. The 386 could do everything the 286 could do, and it could do it faster. The clock speed of the 286 ranged from 6 to 12 MHz, whereas the 386 clock speed ranged from 16 MHz to 33 MHz and beyond.

As well as greater speed, the 386 offered some major new features:

- A 386 can access 4096MB of RAM. Yes, four billion bytes. Even by today's standards, that's a b-i-g spreadsheet.
- Another innovation introduced in the 386 was that extended memory, with some help from software drivers, could behave and act like expanded memory. This allowed 386 users with lots of memory to take advantage of this memory in applications written to use expanded memory.
- Probably the most significant new technology introduced in the 386 was the ability of the microprocessor to run in *Virtual 8086 Mode*. In this mode, the 386 supports multiple 8086 sessions — in other words, the system can be configured to use extended memory to emulate multiple 640K base memory areas. Each session appears to the software as though it is running in Real Mode. Products such as DESQview 386 and Windows 3.*x* in Enhanced Mode take advantage of this capability, allowing you to run multiple DOS sessions concurrently.
- The unused memory blocks in the upper memory area can be filled with extended memory. DOS can then access this memory to store itself, TSRs, and device drivers, freeing precious base memory for applications software.

## The Protected-Mode Addressing Scheme

Just like the 8086, the 286 microprocessor uses two 16-bit registers for storing memory addresses. You may recall that the 8086 uses one register to store the memory segment and the other to store the memory offset. In Protected Mode, a similar scheme is used, but the segment register is used to store an index into a *segment descriptor table.* The descriptor table, which is stored in memory, contains a list of 24-bit linear memory addresses. A specific linear memory address is obtained by looking up the 24-bit segment in the descriptor table and adding the offset to it.

In addition to the linear segment address, the descriptor table includes a description of the type of information stored in memory — whether the data is program code or program data, whether the data is read-only or can be modified, and so on.

The 386 and 486 microprocessors offer a further advancement. The same basic addressing scheme is employed; however, the segment and offsets use 32-bit registers, providing access to $2^{32}$ bytes or 4096MB of memory. That's a huge warehouse.

The next generation of operating systems is bypassing the 286 Protected-Mode scheme and going directly to the 32-bit system. Windows NT and OS/2 3.0 are two prime examples.

All these new features essentially divided the user community into two camps: those who have 386 systems and those who don't. More and more programs and operating systems that run only on 386 or later systems are being developed.

After the 386, Intel introduced the 486 microprocessor. The 486 offers a faster clock speed than its predecessor, a built-in math coprocessor, and numerous other technological advances, but in terms of memory management, the 486 is the same as the 386.

## SX, DX, and DX2 Microprocessors

During the late 1980s, computer-purchasing habits changed. Instead of buying higher priced PCs from the main vendors such as IBM and Compaq, consumers switched to budget-priced PCs. In many cases, people simply compared prices of comparable PCs and bought the cheapest one.

To provide a broader price spectrum of microprocessors and to encourage users to buy 386 computers rather than 286s, Intel introduced the 80386SX. To avoid confusion between the 80386 and the 80386SX, Intel added a DX suffix to the standard 80386 processor. So now we have the 80386SX and the 80386DX.

Internally, the 80386SX is the same as the 80386DX, and software that runs on one chip also runs on the other. The difference lies in the way the microprocessor communicates with the other devices in the system. The standard 80386DX uses a 32-bit bus interface, whereas the 80386SX uses a 16-bit interface. Because the 8086 and 80286 chips used 16-bit interface, the supporting circuitry is cheaper for a 16-bit microprocessor. Manufacturers can therefore produce 80386SX computers for less than 80386DX systems. The down side is that an 80386SX system is slower than an 80386DX system. You get what you pay for.

Continuing the theme of offering microprocessor price/performance options, Intel introduced the i486DX and i486SX. The *i* prefix in the name indicates that the product is manufactured by Intel and is used to differentiate Intel chips from those produced by other vendors. With 486 microprocessors, the difference between the DX and the SX system is not the bus interface. Both systems use a 32-bit interface. Believe it or not, the i486SX is exactly the same as the i486DX, but the on-board math coprocessor is disabled. Yes, the SX chip costs the same to manufacture as the DX, but it is sold for less. Imagine disabling the anti-lock brakes on a car and selling it for less. Marketing madness.

Along with the standard 486 chips, Intel offers an *overdrive* option. Chips in the overdrive family internally process instructions at twice the clock speed. In terms of overall throughput, a system equipped with overdrive runs approximately 50 percent faster than it does with the standard chip. It may even handle some numeric operations many times faster than that.

When the overdrive option was first introduced for i486SX systems, a second processor (or overdrive) chip had to be installed on the system board next to the i486SX chip. Later, an i486DX2 chip designed to replace existing i486DX chips was introduced. If you have a PC with an i486DX microprocessor, you can probably significantly upgrade your system by pulling out the i486DX chip and replacing it with the i486DX2.

## High memory

The 286 and later processors can access extended memory by switching the microprocessor into Protected Mode. DOS, however, cannot run in Protected Mode, so the extra memory doesn't help standard DOS applications.

Thanks to a quirk in the 286 chip design, almost the entire first 64K of extended memory can be accessed without switching into Protected Mode. In other words, DOS in Real Mode can access nearly 64K extra bytes on systems with a 286 processor. Although 64K doesn't sound like much, it's an extra 10 percent that's well worth having. In fact, to qualify as a DOS Jedi, you *must* use it.

The first 64K of extended memory is called the *high memory area,* or HMA. Quarterdeck first exploited this area of memory; DOS has supported it directly since version 5.0. You use the DOS device driver HIMEM.SYS to make this high memory available to DOS. Although mainstream applications such as Lotus 1-2-3 and WordPerfect cannot directly use the HMA, you can load the DOS kernel into the HMA. Doing so frees up as much of the base 640K as possible. Windows can also take advantage of the HMA.

### How the HMA Is Addressed

The 286 and later computers use a 20-bit memory address, providing access to $2^{20}$ or 1MB of memory. Twenty pins, numbered A0 to A19, connect the CPU to the memory bus. The twenty-first pin, A20, is used as a toggle switch to indicate whether the CPU is in Real or Protected Mode. If the CPU is in Real Mode and a program tries to access memory beyond the 1MB range, the address is wrapped around to the low memory. It's like a 12-hour digital clock: After you reach the time 11:59, the value wraps around to 00:00. On the other hand, a 24-hour clock moves to 12:00 and beyond — you might call this a Protected-Mode clock!

By twiddling with the A20 pin, you can trick DOS into accessing the high memory area. With the A20 pin high, DOS actually thinks that it is addressing FFFF:0010 to FFFF:FFFF, which is 16 bytes short of 64K, but it is actually addressing the memory from 10000:0000 to 10000:FFEF.

As you can imagine, manipulation of the A20 pin is very hardware dependent, and some systems respond differently than others. HIMEM.SYS tries to determine the precise hardware model and then configures itself accordingly. You can use the /MACHINE switch to specify the hardware when HIMEM.SYS doesn't recognize the system.

## Other memory terms

The following section provides brief descriptions of other memory terms you may encounter in software manuals or magazines.

### Memory managers

Memory managers are utility programs that are responsible for managing the diverse memory needs of applications software. Typically, they control how installed memory is configured — that is, whether it is expanded or extended, where software is loaded, and how memory is allocated to applications when they request it.

Memory management products usually include more than one program. For example, one program may be responsible for expanded and extended memory allocation, and another for managing the HMA. Memory managers often include a program that analyzes all the software you installed and adjusts the CONFIG.SYS and AUTOEXEC.BAT files to help make optimum use of memory.

Two of the most popular third-party memory managers are Quarterdeck's Expanded Memory Manager and Qualitas' 386MAX. Many of the features offered in these products are now available in DOS 6.

### DOS extenders

Writing Protected-Mode applications is complicated. Believe me — I've done it. To shield the programmer from the nitty gritty complications of writing Protected-Mode applications, a number of companies that specialize in programming tools release libraries of Protected-Mode routines. These program libraries provide a layer between the application program and DOS and are known as *DOS extenders.* The programmer makes simple calls to the library, and the DOS extender makes the low-level calls to the operating system or hardware.

Two of the better known libraries are DOS-Extender from Phar Lap software and DOS/16M from Rational Systems. As a DOS user, you wouldn't ordinarily care about these behind-the-scenes tools, but occasionally a program written with one of these libraries may generate a DOS extender error message.

The industry developed a set of programming standards called the *DOS Protected Mode Interface,* or DPMI. Most Protected-Mode applications now comply with DPMI version 0.9. These standards ensure that different applications can cooperatively use Protected Mode. For example, Windows 3.1 and Lotus 1-2-3 Release 3 can run simultaneously, thanks to the DPMI standard.

One other programming standard you may encounter is called the *Virtual Control Program Interface,* or VCPI. This standard identifies an approach for converting expanded memory back to extended memory, when the expanded memory is really extended memory that has been configured as expanded. *Phew!*

## Upper memory blocks

Upper memory is reserved for video memory, ROM-BIOS, and other hardware memory, but much upper memory goes unused. The 386 and later systems can map extended memory into these unused areas and store device drivers and other TSRs in them. The areas of extended memory mapped into unused upper memory are referred to as *upper memory blocks,* or UMBs.

## Shadow RAM

In general, ROM is much slower then RAM. When DOS needs to access a disk drive, write to video memory, or access many other hardware services, it must execute code in the ROM-BIOS. To speed up these operations, some PCs implement a system known as *shadow RAM.* When the computer first boots, a copy of ROM-BIOS is transferred to RAM, which is a lot faster than ROM. When DOS tries to access ROM-BIOS, it is diverted into reading the copy of ROM in extended memory.

## Stealth and BIOS compression

In the battle for RAM-squeezing supremacy, some of the leading memory management vendors developed ingenious techniques to find every last byte of unused upper memory.

QEMM-386 v6.0, from Quarterdeck Office Systems, uses a special ROM-switching technique known as *Stealth.* Stealth hides ROM-BIOS, video ROM, and even disk ROM, and fills the vacated addresses with UMBs. This approach is very machine-dependent and does not work on all systems.

386MAX, from Qualitas, identifies unused areas of BIOS and converts it to UMBs. This technique, called *BIOS compression,* can yield up to 32K extra bytes on Compaq and IBM systems.

## Virtual memory

Virtual memory provides a way for applications to access more memory than is physically installed in the PC. If all the physical memory is already being used and an application requests more memory, a virtual memory manager can temporarily store some of the data held in memory in a file on disk, thereby freeing up that memory area for the new application. When the owner of the disk-based memory tries to access its memory area, the data must be swapped back from disk to RAM.

As you might imagine, virtual memory is much slower than RAM because it is stored on disk. However, if you don't have the memory, slow is better than stop.

Windows is the best-known application that supports virtual memory. The infamous swap file serves as temporary storage for virtual memory.

# How to Maximize Memory with DOS 6

Now that you have learned the theory, it's time to put your knowledge to practical use. In this section, you learn how to use DOS commands such as HIMEM.SYS, EMM386.EXE, and MEMMAKER to maximize your memory utilization.

## Reviewing the DOS 6 memory arsenal

DOS 6 provides some first-class memory management tools. Table 5-1 provides a brief description of each of the memory management tools available in the DOS 6 arsenal.

The amount of memory you can utilize depends on the type of microprocessor and the type and quantity of memory installed in your system.

**Table 5-1 DOS Memory Management Tools**

| ***DOS Tool*** | ***Use*** |
|---|---|
| HIMEM.SYS | The extended memory manager, responsible for policing the use of extended memory; among other things, HIMEM.SYS enables you to load DOS *high* (into the HMA) |
| EMM386.EXE | Uses extended memory to simulate expanded memory and allows extended memory to be used as UMBs for loading programs and device drivers |
| DOS=HIGH | Instructs DOS to load itself into the HMA |
| DOS=UMB | Instructs DOS to manage the programs in the UMBs |
| MEM.EXE | Displays the status of installed memory, including the location of programs in memory and the amount of memory programs are using |
| MSD.EXE | Displays diagnostic information about the PC's hardware, BIOS, and software |
| DEVICEHIGH | Loads device drivers into the UMBs |
| LOADHIGH | Loads programs into the UMBs |
| MEMMAKER | Assesses your system's memory utilization and modifies CONFIG.SYS and AUTOEXEC.BAT to make better use of available memory |

Don't forget, you can use the MSD.EXE program to determine your microprocessor type and your installed memory (see the section "How to check your system" earlier in the chapter).

## Full steam ahead with the 80386

If you have a 386 or later computer system, you can take full advantage of all the memory tools available in DOS 6.

These tools help you maximize base memory by filling unused upper memory with extended memory (UMBs), into which you can then load device drivers and terminate-and-stay-resident programs (TSRs). In Real Mode, you can access the first 64K of extended memory as high memory (HMA) and then load DOS into HMA. On 386+ systems, EMM386 can also use extended memory to emulate expanded memory.

## Options for 286 systems

Many of the DOS 6 memory management features take advantage of hardware introduced in the 386 microprocessor. For example, the primary management tool, EMM386, runs on 386+ systems only, as its name implies. Without EMM386, there is no support for UMBs. MEMMAKER's primary function is to optimize the use of UMBs, and so it, too, only runs on 386+ systems.

However, 286 systems such as IBM PC ATs and compatibles do support extended memory and the HMA. You can therefore use HIMEM.SYS to manage the HMA and load DOS high. But that's all, folks.

## Bad news for 8086 owners

If you own an 8086 system such as the original IBM PC or IBM PC XT, I've got some bad news for you. You cannot create UMBs, and you do not have access to the HMA because your computer can't access extended memory. In other words, none of the memory-increasing tools provided with DOS 6 work with 8086 systems.

## Alternatives for 8086 and 286 owners

Although DOS 6 shortchanges 8086 and 286 owners as far as memory management capabilities are concerned, a number of third-party products do provide useful memory management tools. These products provide many of the UMB facilities by exploiting the capabilities of specific 80286 chip sets, such as NEAT, AT/386, and LEAP CHIPSet from Chips and Technologies. Many third-party memory management tools also can utilize LIM-EMS 4.0 expanded memory hardware. Two of the most popular products are QRAM from Quarterdeck Office Systems and MOVE'EM from Qualitas. Both QRAM and MOVE'EM bring UMB capabilities to these older systems.

Bob's Shareware Collection disks include the excellent memory manager UMB_DRVR.SYS from Christopher Blum. UMB_DRVR.SYS is a versatile memory device driver, which can use the shadow RAM on many of the 286 chip sets and can provide more than 200K of upper memory on 286 systems.

## Reviewing your memory settings with MEM

The process of fine-tuning your PC's memory configuration involves some degree of trial and error. You look at the memory available, change some settings, reboot the system, and see whether you increased the free memory.

You can use the DOS 6 command MEM to review your memory usage. The amount and type of detail displayed by MEM depends on which switches you use. If you run MEM without any switches, the display will be a brief summary, as follows:

```
Memory Type        Total  =  Used   +  Free
-------------------------------------------
Conventional        640K       87K      553K
Upper                59K       59K        0K
Adapter RAM/ROM     325K      325K        0K
Extended (XMS)    15104K    14080K     1024K
-------------------------------------------
Total memory      16128K    14551K     1577K

Total under 1 MB    699K      146K      553K

EMS is active.
Largest executable program size       553K  (565952 bytes)
Largest free upper memory block         0K       (0 bytes)
MS-DOS is resident in the high memory area.
```

You also can use the following switches with the MEM command:

| | |
|---|---|
| /C | Shows how much base and upper memory is being used by each active program and device driver and displays a summary of overall memory usage. The long form of this switch is /CLASSIFY. |
| /D | Shows detailed information about each active program and device driver, including the item's size, segment address, and memory type. The long form of this switch is /DEBUG. |
| /F | Displays the free base and upper memory regions. The long form of this switch is /FREE. |
| /M | Provides complete memory information about a specific program. You must specify the name of the program or device driver following the /M switch. The long form of this switch is /MODULE. |

/P Instructs MEM to pause when the display is filled. (In other words, it stops MEM output from scrolling off the screen.) To restart the display, you press any key. You won't be surprised to hear that the long form of this command is /PAUSE.

The DOS 5 version of MEM does not support the same switches as DOS 6. In DOS 5, /P is an abbreviation for /PROGRAM, which instructs MEM to display detailed information about each program loaded in memory. On DOS 5 systems, use the |MORE pipe as an alternative to /PAUSE. The /C and /D switches are supported, but in general, the MEM command in DOS 6 provides more information than its predecessor.

For a full description of the MEM switches, refer to Part IV, "The Complete DOS 6 Command Reference."

When I tune a PC's memory, I primarily use the /C and /P switches. If a program is experiencing memory problems, I use the /D switch to obtain more detailed information. The following listing shows the output from the MEM command when the /C switch is specified:

```
Modules using memory below 1 MB:

  Name         Total       =   Conventional      +   Upper Memory
  ----------------------------------------------------------------
  MSDOS       18605   (18K)        18605   (18K)           0    (0K)
  HIMEM        1200    (1K)         1200    (1K)           0    (0K)
  EMM386       3072    (3K)         3072    (3K)           0    (0K)
  SETVER        624    (1K)          624    (1K)           0    (0K)
  SATISFAX     4032    (4K)         4032    (4K)           0    (0K)
  SMARTDRV    30832   (30K)        30832   (30K)           0    (0K)
  PROTMAN       128    (0K)          128    (0K)           0    (0K)
  COMMAND      3088    (3K)         3088    (3K)           0    (0K)
  win386      16096   (16K)        12112   (12K)        3984    (4K)
  WIN          1664    (2K)         1664    (2K)           0    (0K)
  SHARE        5248    (5K)         5248    (5K)           0    (0K)
  CASMGR       5504    (5K)         5504    (5K)           0    (0K)
  COMMAND      3360    (3K)         3360    (3K)           0    (0K)
  MOUSE       17120   (17K)            0    (0K)       17120   (17K)
  MVSOUND      9184    (9K)            0    (0K)        9184    (9K)
  TSLCDR      12768   (12K)            0    (0K)       12768   (12K)
  EXP16        8976    (9K)            0    (0K)        8976    (9K)
  WORKGRP      4400    (4K)            0    (0K)        4400    (4K)
  DOSKEY       4144    (4K)            0    (0K)        4144    (4K)
  Free       565968  (553K)       565968  (553K)           0    (0K)
```

```
Memory Summary:

  Type of Memory       Size       =     Used       +     Free
  ---------------------------------------------------------------------
  Conventional      655360   (640K)      89392    (87K)    565968   (553K)
  Upper              60576    (59K)      60576    (59K)         0     (0K)
  Adapter RAM/ROM   332640   (325K)     332640   (325K)         0     (0K)
  Extended (XMS)  15466496 (15104K)   14417920 (14080K)   1048576  (1024K)
  ---------------------------------------------------------------------
  Total memory    16515072 (16128K)   14900528 (14551K)   1614544  (1577K)

  Total under 1 MB  715936   (699K)     149968   (146K)    565968   (553K)

  EMS is active.
  Largest executable program size      565952   (553K)
  Largest free upper memory block           0     (0K)
  MS-DOS is resident in the high memory area.
```

## Using the big gainers

There are some very simple ways of dramatically increasing the amount of available base memory on a 386 system. Before worrying about squeezing out 1K of memory here and another 2K there, you should take advantage of the "big gainers."

PC memory is configured and controlled by the start-up files CONFIG.SYS and AUTOEXEC.BAT. You can realize big gains in memory usage by making a few simple changes in CONFIG.SYS.

### Load DOS high

The first item of business is to take advantage of the HMA capabilities and load DOS high. This is the only DOS 6 feature available to 286 users. To load DOS high, you must install HIMEM.SYS. You then can instruct DOS to load itself high by adding the statement DOS=HIGH in CONFIG.SYS. If DOS is loaded high, the BUFFERS also are loaded high.

The following typical CONFIG.SYS file illustrates this technique:

```
FILES=50
BUFFERS=15
DEVICE=C:\WINDOWS\MOUSE.SYS /Y
DEVICE=C:\DOS\SMARTDRV.EXE /DOUBLE_BUFFER
```

When I boot my PC with the preceding plain CONFIG.SYS, MEM indicates that there are approximately 571,000 bytes of base memory free. That's not much memory, considering that only DOS, a disk cache, and a mouse driver are loaded. MEM shows that the memory is consumed by these programs as follows:

```
Name             Total      =    Conventional    +   Upper Memory
---------------------------------------------------------------------
MSDOS          60093    (59K)       60093    (59K)        0     (0K)
MOUSE          17088    (17K)       17088    (17K)        0     (0K)
SMARTDRV        2480     (2K)        2480     (2K)        0     (0K)
COMMAND         5024     (5K)        5024     (5K)        0     (0K)
Free          570704   (557K)      570704   (557K)        0     (0K)
```

After the CONFIG.SYS file is modified to load DOS high, it appears as follows:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\WINDOWS\MOUSE.SYS /Y
DEVICE=C:\DOS\SMARTDRV.EXE /DOUBLE_BUFFER
BUFFERS=15,0
FILES=50
DOS=HIGH
```

Just by adding the HIMEM.SYS and DOS=HIGH statements, the free memory jumps to nearly 616,000 bytes, and MEM reports the following memory usage:

```
Name             Total      =    Conventional    +   Upper Memory
---------------------------------------------------------------------
MSDOS          15757    (15K)       15757    (15K)        0     (0K)
HIMEM           1200     (1K)        1200     (1K)        0     (0K)
MOUSE          17088    (17K)       17088    (17K)        0     (0K)
SMARTDRV        2480     (2K)        2480     (2K)        0     (0K)
COMMAND         2960     (3K)        2960     (3K)        0     (0K)
Free          615920   (601K)      615920   (601K)        0     (0K)
```

Notice that when DOS is loaded high, a 15K stub is left in base memory.

## Use the UMBs

You may recall that upper memory is designed for video memory, BIOS, and plug-in hardware extensions, but that much of the upper memory often goes unused. The second big gainer is to allocate some of the system's extended memory to fill the unused portions of upper memory — creating upper memory blocks, or UMBs for short.

You can use the EMM386.EXE program to create UMBs by adding an EMM386 statement to CONFIG.SYS. EMM386 performs many functions, and the RAM switch must be used to instruct EMM386 to convert extended memory to UMBs.

Having installed EMM386, you can instruct DOS to manage the UMBs by changing the DOS= statement in CONFIG.SYS to DOS=HIGH,UMB.

If you are using a third-party memory management program instead of EMM386.EXE, you usually should not use the DOS=UMB statement. The memory manager will manage the UMB space independently.

When you review existing CONFIG.SYS files, you may find two different DOS= statements. One line will say DOS=HIGH and the other DOS=UMB. These two separate statements have the same effect as the single statement DOS=HIGH,UMB. These separate statements are often created by the DOS memory optimizer, MEMMAKER (discussed later). MEMMAKER adds a second DOS= statement if it finds that the UMB component is missing.

Having created UMBs, you can now take advantage of them by loading device drivers and programs into the UMBs. Programs loaded into UMBs are said to be *loaded high*. The DOS commands DEVICEHIGH and LOADHIGH are used to load device drivers and programs high, respectively. In the example configuration, the only device driver being loaded is MOUSE.SYS, but it can be loaded into the UMBs simply by changing the DEVICE statement to DEVICEHIGH.

The example CONFIG.SYS file is therefore updated to utilize UMBs as follows:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DEVICE=C:\DOS\SMARTDRV.EXE /DOUBLE_BUFFER
BUFFERS=15,0
FILES=50
DOS=HIGH,UMB
DEVICEHIGH=C:\WINDOWS\MOUSE.SYS /Y
```

The effects of these changes on free memory are summarized by MEM as shown in the following listing:

```
Name         Total    =   Conventional   +  Upper Memory
-------------------------------------------------------------
MSDOS        15757   (15K)    15757   (15K)       0     (0K)
HIMEM         1200    (1K)     1200    (1K)       0     (0K)
EMM386        3072    (3K)     3072    (3K)       0     (0K)
SMARTDRV      2480    (2K)     2480    (2K)       0     (0K)
COMMAND       2960    (3K)     2960    (3K)       0     (0K)
MOUSE        17120   (17K)        0    (0K)   17120    (17K)
Free        706080  (690K)   629920  (615K)   76160    (74K)
```

```
Memory Summary:

  Type of Memory         Size      =       Used      +       Free
  ----------------  ----------------   ---------------   ------------------
  Conventional        655360   (640K)     25440    (25K)     629920   (615K)
  Upper                93280    (91K)     17120    (17K)      76160    (74K)
  Adapter RAM/ROM     299936   (293K)    299936   (293K)          0     (0K)
  Extended (XMS)    15466496 (15104K)    589824   (576K)   14876672 (14528K)
  ----------------  ----------------   ---------------   ------------------
  Total memory      16515072 (16128K)    932320   (910K)   15582752 (15218K)
```

As you can see, EMM386 consumes 3K of base memory, but it is more than offset by the 17K saved by loading the mouse driver into UMBs. What's more, an additional 74K of UMBs are available for other device drivers and programs like SHARE, network shells, DOSKEY, etc.

## Shifting from RAM cram to UMB cram

In the 8086/286 days, everybody complained about RAM cram. With DOS, TSRs like Sidekick, and large programs, there was never enough RAM. Along came UMBs, and suddenly RAM cram was a thing of the past.

If you ever moved into a larger apartment or house, you might remember wondering what you were going to do with all the space. But it wasn't long before you filled the place up and started complaining about the lack of room. The same goes for UMBs. Now people complain about UMB cram — too many programs and too little UMB space in which to put them.

Making the best use of the available UMBs memory is not as straightforward as you might think.

### Identifying potential UMB space

The upper memory area is a memory mine field — here's ROM-BIOS, there's empty space, over there is video memory, and down there is another empty space. To complicate things even more, the organization of upper memory varies from system to system. Figure 5-7 shows how upper memory is organized on the most common systems. As a general rule, ROM-BIOS is located in the F000-FFFF range, the range A000-BFFF is used for video memory, and the C000-CFFF range is used for video and disk BIOS.

EMM386 is responsible for mapping extended memory into the unused portions of upper memory. You can instruct EMM386 to include or exclude specific areas of upper memory with the I= and X= directives, respectively.

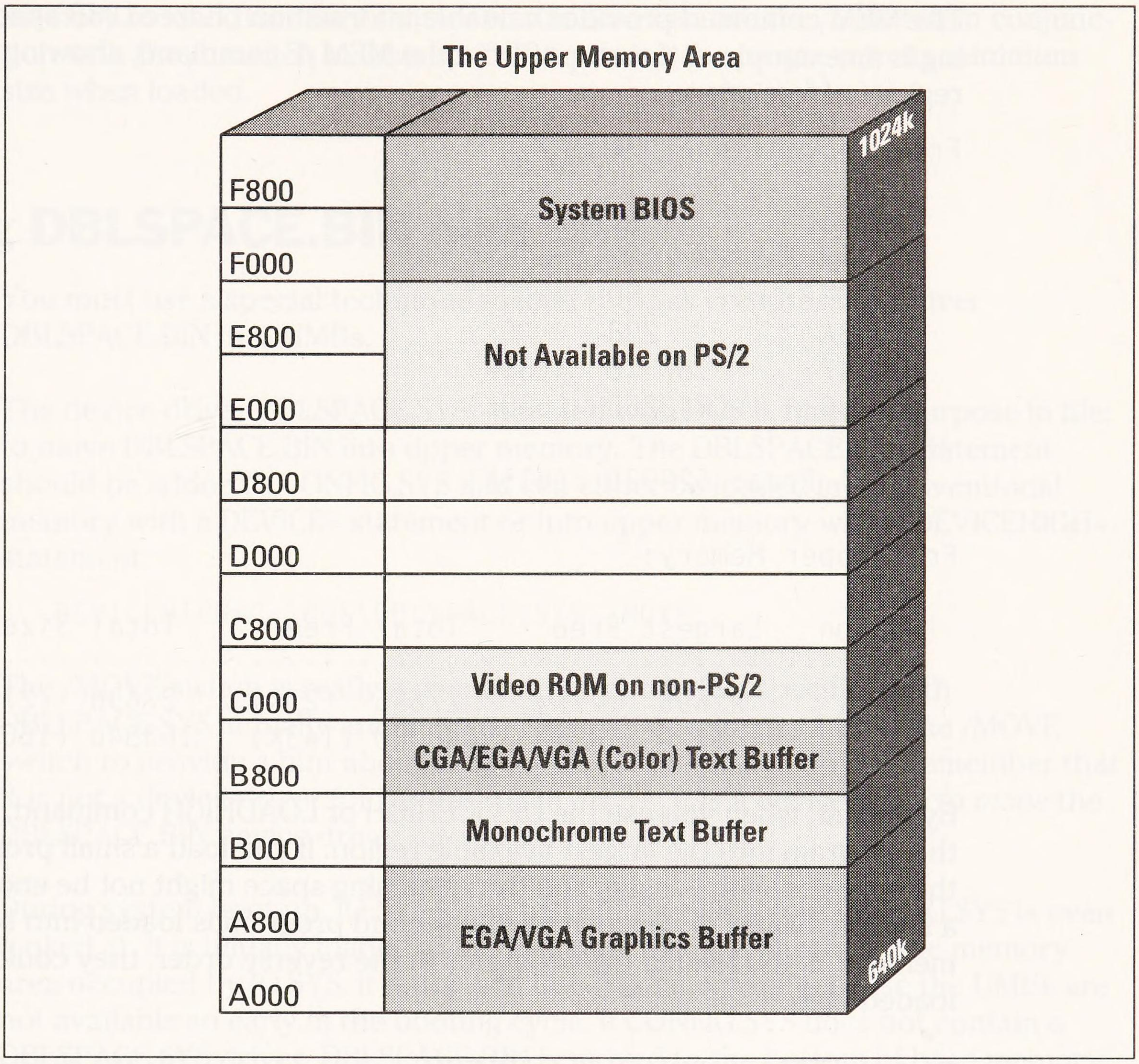

The Upper Memory Area
1024k
F800
System BIOS
F000
E800
Not Available on PS/2
E000
D800
D000
C800
Video ROM on non-PS/2
C000
CGA/EGA/VGA (Color) Text Buffer
B800
Monochrome Text Buffer
B000
A800
EGA/VGA Graphics Buffer
A000
640k

**Figure 5-7:** How upper memory is utilized on popular systems.

By default, the DOS 5 version of EMM386 only supported upper memory in the range C000-DFFF. The I= switch had to be used to instruct EMM386 to access the other areas. Many books and magazines recommended using the switch I=E000-EFFF to get an extra 64K of UMB space. Starting with DOS 6, EMM386 now supports the range E000-EFFF by default, so you no longer need to explicitly include it.

## Controlling load order and location

Because UMBs are mapped into the unused portions of upper memory, they are fragmented, with some blocks of memory at one address and other blocks of memory at another. These different areas of UMBs are referred to as *regions*.

The MEM command provides valuable information on free UMB space. Following is an example of the output from the MEM /F command, showing 2 UMB regions of free space:

```
Free Conventional Memory:

  Segment          Size
  --------------------------
   00624             80    (0K)
   0063A            208    (0K)
   00647          87376   (85K)
   01B9C         542256  (530K)

  Total Free: 629920  (615K)

Free Upper Memory:

  Region    Largest Free     Total Free       Total Size
  -----------------------------------------------------
      1      27696  (27K)    27696  (27K)     27696  (27K)
      2    146720 (143K)  146720 (143K)   163840 (160K)
```

By default, when you use the DEVICEHIGH or LOADHIGH command, DOS loads the program into the largest available region. If you load a small program first, the largest region is used, and the remaining space might not be enough to load a second, bigger program. If so, the second program is loaded into base memory. If you loaded the programs in the reverse order, they could both be loaded high.

Both the DEVICEHIGH and LOADHIGH commands provide the /L switch so you can specify the specific region(s) where a program is loaded.

By fine-tuning the order in which programs are loaded and by using the /L switch, you can make optimum use of the available UMBs.

## Tweaking UMB memory allocation

Effective UMB management is further complicated by the fact that some programs require more memory when they are loaded, but then shrink their memory requirements after successful installation. It's like a building that needs a whole block during construction but only occupies half a block after it's built. The reverse is also true; some programs require less space to load than they do to operate.

The DEVICEHIGH and LOADHIGH /L switch optionally enable you to identify the minimum amount of memory that must be available in the region for the program to be loaded high. This is necessary for programs that require less memory to load than they require to run. You can use the /S switch in conjunction with the /L switch to instruct DOS to shrink the program to its minimum size when loaded.

## Loading DBLSPACE.BIN high

You must use a special technique to load the disk compression driver DBLSPACE.BIN into UMBs.

The device driver DBLSPACE.SYS, included with DOS 6, has one purpose in life: to move DBLSPACE.BIN into upper memory. The DBLSPACE.SYS statement should be added to CONFIG.SYS and can either be loaded into conventional memory with a DEVICE= statement or into upper memory with a DEVICEHIGH= statement:

```
DEVICEHIGH=C:\DOS\DBLSPACE.SYS /MOVE
```

The /MOVE switch is really a comment. Any switches specified with DBLSPACE.SYS actually are ignored. The DOS developers added the /MOVE switch to provide a hint about the purpose of the device driver. Remember that it is not a device driver for DoubleSpace drives; it is a device drive to *move* the DBLSPACE.BIN device driver into upper memory.

**GEM**

During system boot-up, IO.SYS loads DBLSPACE.BIN before CONFIG.SYS is even looked at. It is initially loaded at the top of memory, adjacent to the memory area occupied by IO.SYS. It must load into base memory because the UMBs are not available so early in the booting cycle. If CONFIG.SYS does not contain a DBLSPACE.SYS driver, DBLSPACE.BIN is moved to the bottom of base memory after all the CONFIG.SYS statements are processed.

## Letting MEMMAKER do the work

Fully optimizing the use of UMBs is a painstaking, complicated, and laborious process. In fact, it's the kind of work a computer should be doing!

Fortunately, DOS engineers agreed and added a brand new program, MEMMAKER, to DOS 6. The sole purpose of MEMMAKER is to analyze your PC's memory usage and reconfigure the CONFIG.SYS and AUTOEXEC.BAT files for optimum memory usage. It's an excellent program.

## Looking at MEMMAKER in action

Listed next are typical CONFIG.SYS and AUTOEXEC.BAT files from a 386 system that is not configured very well:

CONFIG.SYS:

```
FILES=50
BUFFERS=15
DEVICE=C:\WINDOWS\MOUSE.SYS /Y
DEVICE=C:\DOS\SETVER.EXE
LASTDRIVE=M
```

AUTOEXEC.BAT:

```
@ECHO OFF
PATH C:\DOS;C:\;C:\WINDOWS;C:\WINDOWS\NDW;C:\BATFILES
PROMPT $P$G
DOSKEY
SET SAGEEDIT=C:\SPE
SET NU=C:\U\NORTON
SET TEMP=C:\TEMP
```

Using these configuration files, MEM reports 567,520 bytes of free memory as follows:

```
Memory Type        Total  =  Used  +  Free
------------------------------------------
Conventional        640K      86K     554K
Upper                 0K       0K       0K
Adapter RAM/ROM     384K     384K       0K
Extended (XMS)    15104K   15104K       0K
Expanded (EMS)        0K       0K       0K
------------------------------------------
Total memory      16128K   15574K     554K

Total under 1 MB    640K      86K     554K

Largest executable program size       554K  (567520 bytes)
Largest free upper memory block         0K       (0 bytes)
The high memory area is available.
```

After running MEMMAKER in Express Mode, the configuration files have been modified as follows:

CONFIG.SYS:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM WIN=DD00-DFFF WIN=D800-DCFF
BUFFERS=15,0
FILES=50
DOS=UMB
LASTDRIVE=M
FCBS=4,0
DEVICEHIGH /L:1,55168 =C:\WINDOWS\MOUSE.SYS /Y
DEVICEHIGH /L:1,12048 =C:\DOS\SETVER.EXE
```

AUTOEXEC.BAT:

```
@ECHO OFF
PATH C:\DOS;C:\;C:\WINDOWS;C:\WINDOWS\NDW;C:\BATFILES
PROMPT $P$G
LH /L:1,6400 DOSKEY
SET SAGEEDIT=C:\SPE
SET NU=C:\U\NORTON
SET TEMP=C:\TEMP
```

The free memory increases to 583,216 bytes, and MEM shows the following statistics:

```
Memory Type          Total =   Used   +  Free
----------------------------------------------
Conventional          640K      70K      570K
Upper                  59K      21K       38K
Adapter RAM/ROM       325K     325K        0K
Extended (XMS)      15104K   15104K        0K
----------------------------------------------
Total memory        16128K   15521K      607K

Total under 1 MB      699K      92K      607K

EMS is active.
Largest executable program size          570K  (583216 bytes)
Largest free upper memory block           38K   (38464 bytes)
The high memory area is available.
```

The main changes MEMMAKER made to CONFIG.SYS were to add HIMEM.SYS and EMM386 statements, to add a DOS=UMB statement, and to use DEVICEHIGH to load MOUSE.SYS and SETVER.EXE into the upper memory blocks. The only change to AUTOEXEC.BAT was the addition of the instruction to use

LOADHIGH, in its abbreviated form LH, to load DOSKEY into the upper memory blocks.

Notice that the /L switch is used in every instance of DEVICEHIGH and LOADHIGH to control the region as well as the minimum memory required by each program.

Having already learned about the big memory gainers, you may have observed one omission from the modified CONFIG.SYS file. Although HIMEM.SYS was installed, DOS was not loaded into high memory. If the DOS=UMB line is manually changed to DOS=HIGH,UMB, the free memory jumps to 631,648 bytes.

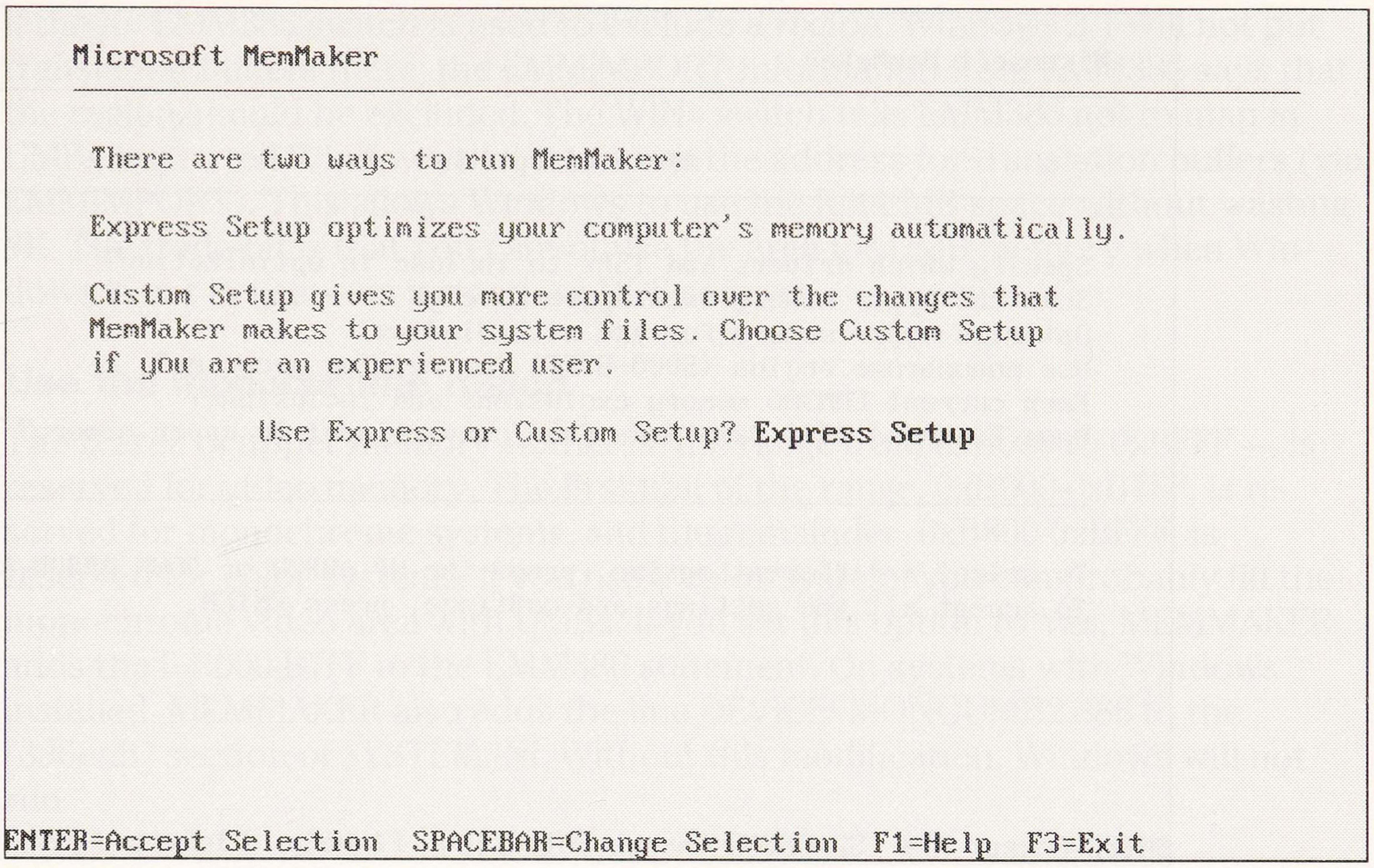
```
Memory Type        Total =  Used  +  Free
-----------------------------------------
Conventional         640K     23K    617K
Upper                 59K     21K     38K
Adapter RAM/ROM      325K    325K      0K
Extended (XMS)     15104K    544K  14560K
-----------------------------------------
Total memory       16128K    913K  15215K

Total under 1 MB     699K     44K    655K

EMS is active.
Largest executable program size       617K  (631648 bytes)
Largest free upper memory block        38K   (38464 bytes)
MS-DOS is resident in the high memory area.
```

You probably wonder why MEMMAKER missed such an obvious improvement. When DOS 6 is first installed, SETUP adds the statement DOS=HIGH to CONFIG.SYS. The MEMMAKER developers assumed that DOS=HIGH must have been removed on purpose, and so MEMMAKER does not add DOS=HIGH if it is missing, even in Custom Mode. Strange.

## Running MEMMAKER

MEMMAKER is one of those all too rare programs that is very powerful *and* easy to use. MEMMAKER is located in the DOS directory and normally is executed without any switches or options. The only keys you need to remember are the spacebar (to toggle menu options), F3 to abort, and Enter to proceed with the selections.

Initially, MEMMAKER displays a welcome screen. After you press a key, the program presents the main option screen (see Figure 5-8), which prompts you to choose between *Express Mode* or *Custom Mode*.



```
Microsoft MemMaker

There are two ways to run MemMaker:

Express Setup optimizes your computer's memory automatically.

Custom Setup gives you more control over the changes that
MemMaker makes to your system files. Choose Custom Setup
if you are an experienced user.

        Use Express or Custom Setup? Express Setup

ENTER=Accept Selection  SPACEBAR=Change Selection  F1=Help  F3=Exit
```

**Figure 5-8:** MEMMAKER can run in Express or Custom Mode.

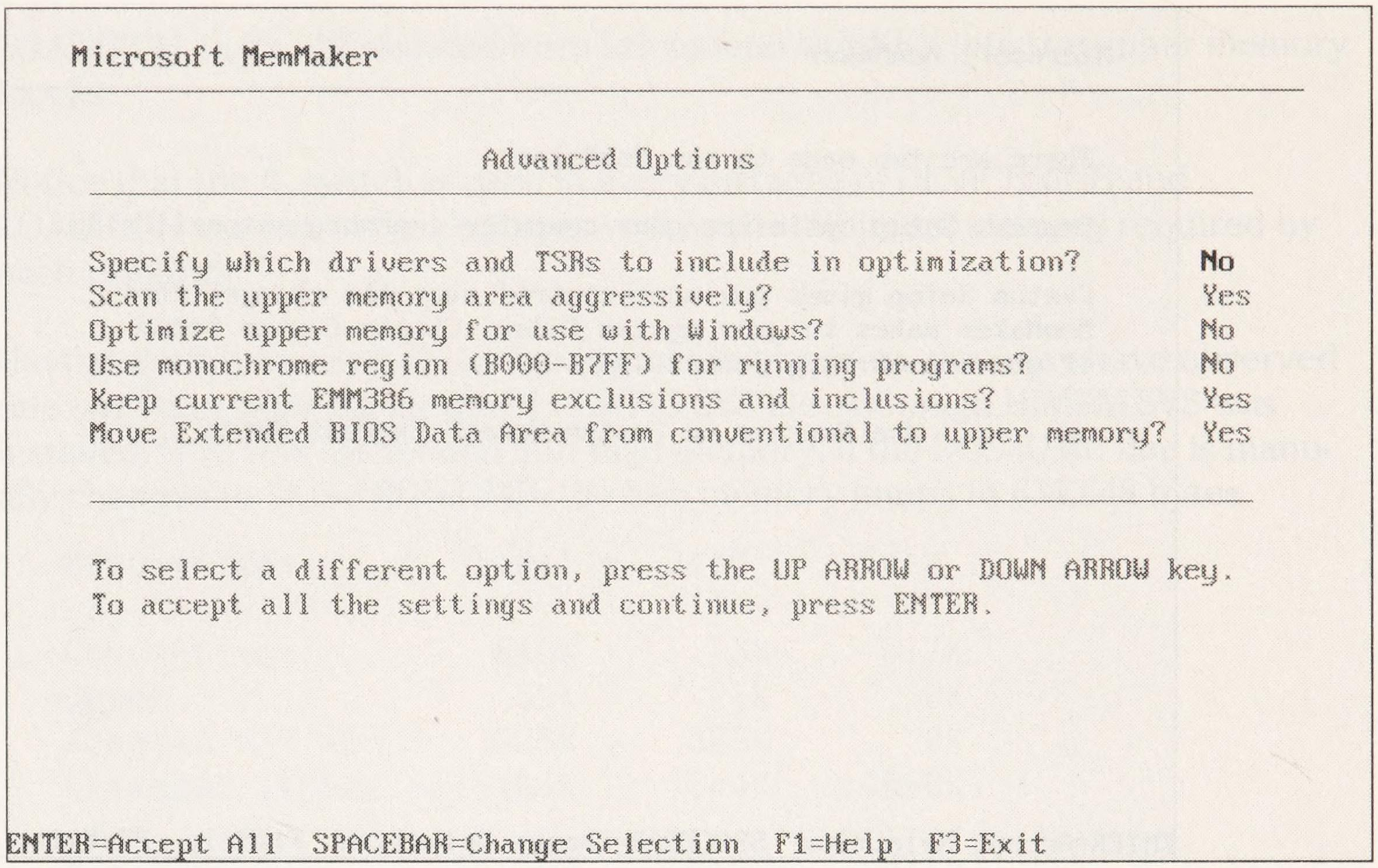
No matter which mode you select, MEMMAKER asks the following question:

```
Do you use any programs that need expanded memory (EMS)?
```

One function of EMM386 is to convert extended memory to expanded memory. As you may recall, expanded memory uses a 64K page frame in upper memory for paging. If you don't run applications that take advantage of EMS, you might as well eliminate the 64K page frame and free that memory for use as UMBs. When you respond No to this question, MEMMAKER adds a NOEMS switch to the EMM386.

After the change to NOEMS is made, certain applications may run much more slowly than they used to. This is a good indication that the application did take advantage of expanded memory after all. Remove the NOEMS switch from the EMM386 statement in CONFIG.SYS, reboot the system, and see whether the applications regain their old speed. If so, you may want to re-run MEMMAKER and respond Yes to the EMS question.

When you select Express Mode, MEMMAKER just gets on with its work. If you choose Custom Mode, you can guide MEMMAKER in some areas of optimization. Figure 5-9 illustrates the options available in Custom Mode. You can use the cursor keys to change the active selection and use the spacebar to toggle the setting between Yes and No.



```
Microsoft MemMaker

                              Advanced Options

  Specify which drivers and TSRs to include in optimization?          No
  Scan the upper memory area aggressively?                            Yes
  Optimize upper memory for use with Windows?                         No
  Use monochrome region (B000-B7FF) for running programs?             No
  Keep current EMM386 memory exclusions and inclusions?               Yes
  Move Extended BIOS Data Area from conventional to upper memory?     Yes

  To select a different option, press the UP ARROW or DOWN ARROW key.
  To accept all the settings and continue, press ENTER.

ENTER=Accept All  SPACEBAR=Change Selection  F1=Help  F3=Exit
```

**Figure 5-9:** Configuration options in the Custom Mode of MEMMAKER.

Unfortunately, MEMMAKER does not scan the existing setup before displaying the custom options. Therefore, the values presented in the configuration options may bear no resemblance to your existing CONFIG.SYS settings.

Chapter 7 of the *MS-DOS User's Guide* provides a good description of most of the custom options, but two options deserve special mention: Optimize upper memory for use with Windows and Use the monochrome region.

### Optimize upper memory for use with Windows

In short, select this option if you use Windows most of the time and are not experiencing low-memory problems when using DOS outside of Windows. MEMMAKER will add one or more WIN= switches to the EMM386.EXE statement in CONFIG.SYS, indicating that the UMB space is reserved for use by Windows.

Windows must create translation buffers in the first megabyte of memory, and, for performance reasons, these translation buffers must be aligned on a 4K boundary. The buffers can, however, be situated in UMBs. Unfortunately, both EMM386 and DOS store information, called *arenas,* at the start and end of each UMB. These arenas cause the start and end of the free memory in the UMB *not* to be 4K-aligned. The first 4K and last 4K of each free UMB space cannot, therefore, be utilized by Windows for translation buffers.

If the X= EMM386 switch is used to exclude a region, Windows 3.1 will not put translation buffers there; the EMMIMPORT information from EMM386 says that the region should be excluded. The WIN= switch tells EMM386 not to map in UMBs but to tell Windows that it can use the address for translation buffers (via EMMIMPORT). This allows Windows to use the full UMB space without wasting 8K. The trade-off is that you cannot use the area for UMB memory when Windows is not in use — in a standard DOS session.

### Use the monochrome region

The B area of upper memory — that is, the range from 0xB000 to 0xBFFF — is reserved for video memory. The first half of the range, 0xB000-0xB7FF, is reserved for monochrome systems, and the remainder, 0xB800-0xBFFF, is reserved for color systems. If you use a color system, you can probably fill the monochrome video area with UMBs. If you set this option to Yes, MEMMAKER adds the I=B000-B7FF to the EMM386 statement. On systems with Windows installed, MEMMAKER also adds the line DEVICE=MONOUMB2.386 to the [386enh] section of SYSTEM.INI. Without this modification, Windows will not run.

TIP

If you are a Windows user and you manually add the I=B000-B7FF statement to the EMM386 statement, make sure that you also edit SYSTEM.INI and add the statement DEVICE=MONOUMB2.386 to the [386enh] section.

In rare circumstances, MEMMAKER may make changes to your configuration files that cause your system to lock up. Refer to the section titled “Solving memory conflicts,” later in the chapter, for guidance on how to proceed.

## Understanding how MEMMAKER works

GEM

In the simple example discussed earlier, MEMMAKER didn’t perform many miracles. However, if you have a complicated configuration with many different device drivers and programs loaded at system boot-up, MEMMAKER can work wonders. MEMMAKER may literally evaluate thousands of configuration permutations before selecting its optimum choice.

The most complicated task MEMMAKER undertakes is determining how much space is required for device drivers and programs, both at load time and thereafter. To perform this feat, MEMMAKER uses two special utilities, CHKSTATE.SYS and SIZER.EXE.

MEMMAKER modifies CONFIG.SYS and adds a new line to the beginning of the file to load the CHKSTATE device driver. If the HIMEM.SYS, EMM386, and DOS=UMB statements are missing, MEMMAKER adds them. MEMMAKER also modifies each DEVICE statement so that SIZER.EXE is used to load and observe the device drivers as they are loaded.

The following listing shows how the example CONFIG.SYS is modified by MEMMAKER during the optimization process:

CONFIG.SYS:

```
DEVICE=C:\DOS\CHKSTATE.SYS /S:SR1 /14277
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
BUFFERS=15,0
FILES=50
DOS=UMB
LASTDRIVE=M
FCBS=4,0
DEVICE=C:\DOS\SIZER.EXE /14277 /8 C:\WINDOWS\MOUSE.SYS /Y
DEVICE=C:\DOS\SIZER.EXE /14277 /9 C:\DOS\SETVER.EXE
```

MEMMAKER also makes modifications to the AUTOEXEC.BAT file. As it does with device drivers, MEMMAKER modifies all program execution statements, including batch files, so that SIZER.EXE is used to load them. At the end of the AUTOEXEC.BAT file, MEMMAKER adds a line to call itself so that the optimization process can proceed.

AUTOEXEC.BAT:

```
@ECHO OFF
PATH C:\DOS;C:\;C:\WINDOWS;C:\WINDOWS\NDW;C:\BATFILES
PROMPT $P$G
C:\DOS\SIZER.EXE /14277 /3 DOSKEY
SET SAGEEDIT=C:\SPE
SET NU=C:\U\NORTON
SET TEMP=C:\TEMP
C:\DOS\MEMMAKER.EXE /SESSION:14277
```

Having edited the configuration files, MEMMAKER reboots the system. Then CHKSTATE.SYS and SIZER.EXE observe and gather statistics as the device drivers and programs are loaded. Using this data, MEMMAKER calculates which programs should be loaded into upper memory, in what order they should be loaded, and how much memory they will need.

Based on these computations, MEMMAKER makes further edits to the configuration files. The main change involves removing the SIZER.EXE statements and replacing them with DEVICEHIGH and LOADHIGH statements, using the appropriate switches. The following listing shows the example configuration files after MEMMAKER made the second set of adjustments:

CONFIG.SYS:

```
DEVICE=C:\DOS\CHKSTATE.SYS /S:FR1 /27107
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM WIN=DD00-DFFF WIN=D800-DCFF
BUFFERS=15,0
FILES=50
DOS=UMB
LASTDRIVE=M
FCBS=4,0
DEVICEHIGH /L:1,55168 =C:\WINDOWS\MOUSE.SYS /Y
DEVICEHIGH /L:1,12048 =C:\DOS\SETVER.EXE
```

AUTOEXEC.BAT:

```
@ECHO OFF
PATH C:\DOS;C:\;C:\WINDOWS;C:\WINDOWS\NDW;C:\BATFILES
PROMPT $P$G
LH L:1,6400 DOSKEY
SET SAGEEDIT=C:\SPE
SET NU=C:\U\NORTON
SET TEMP=C:\TEMP
C:\DOS\MEMMAKER.EXE /SESSION:27107
```

MEMMAKER reboots the system for a second time and then prompts you to see whether everything appears to be working correctly. If you respond Yes, MEMMAKER removes the CHKSTATE statement from the beginning of CONFIG.SYS and removes the MEMMAKER statement from the end of AUTOEXEC.BAT. A summary of your memory gains is displayed, and you are returned to the DOS prompt. You are now ready to take advantage of your newfound memory. If the system appears unstable and you respond No to MEMMAKER's question, the original configuration files can be restored and the system rebooted, leaving the system the same as it was before MEMMAKER started.

Behind the scenes, MEMMAKER creates the file MEMMAKER.STS in the DOS directory. This file contains all the analysis performed by CHKSTATE and SIZER and provides interesting details about each program and device driver loaded at boot-up.

CHKSTATE.SYS writes the summary information at the top of the file, and then SIZER writes the detailed information for each command.

The DOS 6 diagnostic program MSD.EXE can show you how your base and upper memory is used and where programs and device drivers are physically stored in memory. Run MSD and select the **M**emory Block Display option from the Utilities menu. If you scroll through the list of installed programs, the block diagram shows where the program is situated. In Figure 5-10, the location of the WORKGRP.SYS file is identified by the |—| characters in the D400 segment.

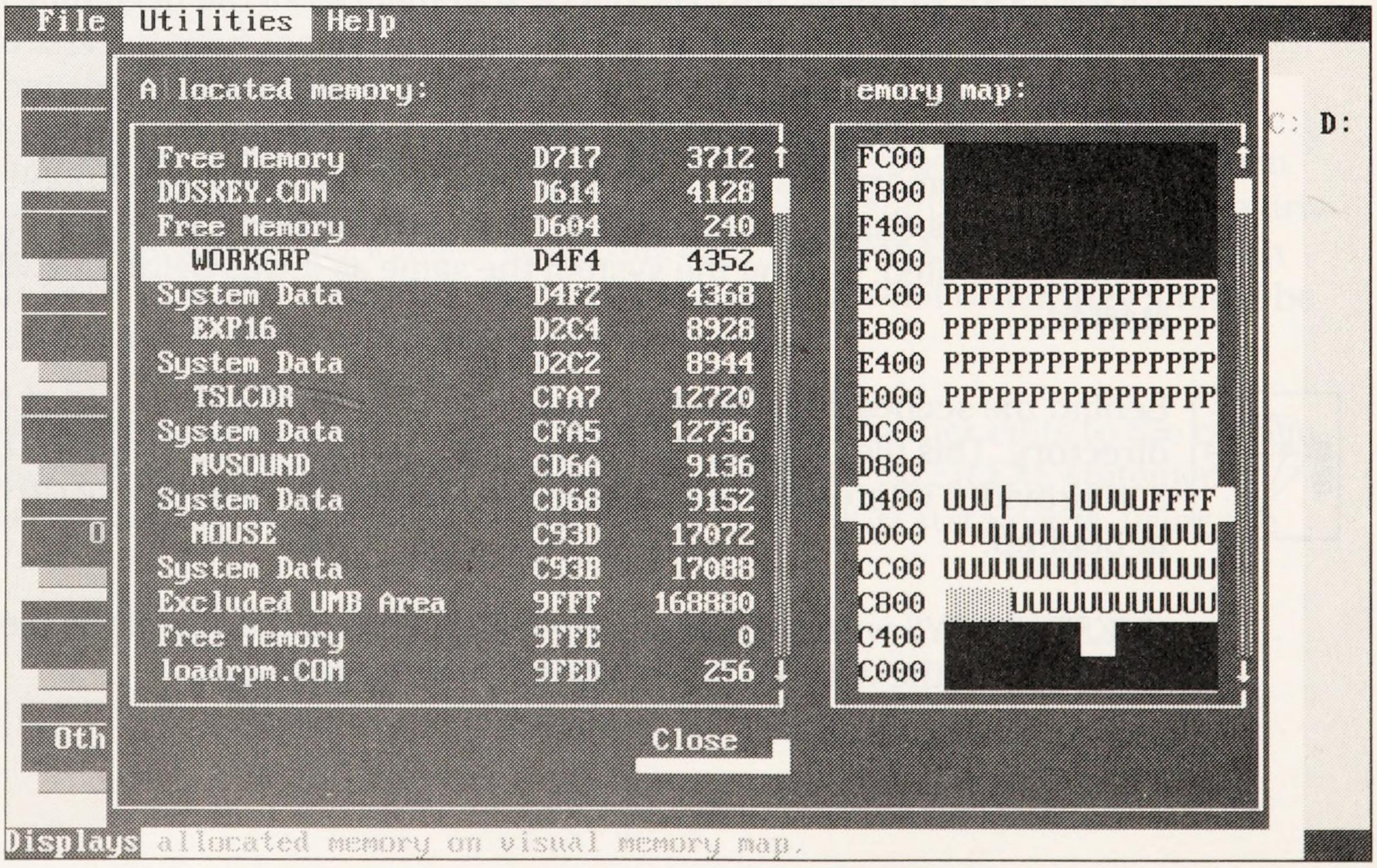
## Helping MEMMAKER succeed

Although MEMMAKER is very "smart," some relatively common situations can reduce its effectiveness. The following section describes some circumstances in which you can give MEMMAKER a helping hand.

### Avoid using a multiple configuration menu in CONFIG.SYS

MEMMAKER is often less effective when the CONFIG.SYS file contains a MENU of options (discussed fully in Chapter 8).

A CONFIG.SYS file that contains more than one configuration may have multiple entries for HIMEM.SYS, EMM386.EXE, etc. When MEMMAKER encounters this situation, it copies the first occurrence of HIMEM.SYS and EMM386.EXE to the beginning of the file. If you instructed MEMMAKER to retain the existing include (I) and exclude (X) switches for EMM386, the switches for the first EMM386 statement are used. Now here's the silly part: MEMMAKER comments out (with a REM statement) all the remaining HIMEM.SYS and EMM386.EXE statements in CONFIG.SYS.



```
Allocated memory:                          Memory map:

Free Memory          D717      3712        FC00
DOSKEY.COM           D614      4128        F800
Free Memory          D604       240        F400
  WORKGRP            D4F4      4352        F000
System Data          D4F2      4368        EC00 PPPPPPPPPPPPPPPP
  EXP16              D2C4      8928        E800 PPPPPPPPPPPPPPPP
System Data          D2C2      8944        E400 PPPPPPPPPPPPPPPP
  TSLCDR             CFA7     12720        E000 PPPPPPPPPPPPPPPP
System Data          CFA5     12736        DC00
  MVSOUND            CD6A      9136        D800
System Data          CD68      9152        D400 UUU|———|UUUUFFFF
  MOUSE              C93D     17072        D000 UUUUUUUUUUUUUUUU
System Data          C93B     17088        CC00 UUUUUUUUUUUUUUUU
Excluded UMB Area    9FFF    168880        C800     UUUUUUUUUUUU
Free Memory          9FFE         0        C400
loadrpm.COM          9FED       256        C000

Displays allocated memory on visual memory map.
```

**Figure 5-10:** MSD shows where device drivers and programs are loaded in upper memory.

In some situations, you may want to have multiple configurations optimized by MEMMAKER. For example, you may have one configuration that doesn't load network drivers, another that loads a scanner driver, and maybe a third configuration that loads everything. On my system, that configuration is called *Enchilada.*

MEMMAKER can configure all the multiple options, but not automatically. The best approach is to make backup copies of CONFIG.SYS and AUTOEXEC.BAT and then create slimmed-down versions of them using the statements for one specific configuration or menu option. Then run MEMMAKER to optimize the configuration for that specific setup. After you do this, merge the modified AUTOEXEC.BAT and CONFIG.SYS files into the backup configuration files, replacing the original statements. Repeat this process for each configuration scenario.

TIP

If you have a device driver or program (such as DOSKEY) that is used in each configuration option, avoid placing the program statement in a [COMMON] section of CONFIG.SYS. Rather, place a copy of the statement in each individual configuration set. Remember that the DEVICEHIGH/LOADHIGH switches may be different for each configuration. Although the program is common to each configuration, the way the program is loaded high may differ depending on the other programs sharing the UMBs.

### Don't call programs from other batch files

In the AUTOEXEC.BAT file, avoid calling other batch files that in turn load programs. For example, don't load DOSKEY and SHARE from a batch file that is itself called from AUTOEXEC.BAT. MEMMAKER analyzes all the programs loaded from the batch file as a single entity. Either all the programs are loaded high or none are loaded high. The solution is to modify AUTOEXEC.BAT before you run MEMMAKER. In this example, edit AUTOEXEC.BAT to call the DOSKEY and SHARE programs directly.

### Don't call programs in IF statements

Avoid running a program from an IF statement in AUTOEXEC.BAT. MEMMAKER will not optimize a program that is called directly from an IF statement. For example, MEMMAKER will ignore the call to DOSKEY in the following statement:

```
if "%anarkey%"=="" doskey
```

Use a GOTO statement in the IF clause, and MEMMAKER will optimize the program's memory usage. This example can be modified as follows:

```
if %anarkey%"=="" goto loaddoskey
.....
:LOADDOSKEY
doskey
.....
```

### Exclude some programs from analysis and execution

You may have some programs and device drivers that you don't want MEMMAKER to analyze — you want them to be executed as is, without any changes to their switches and so forth. There may be other programs that you do not want MEMMAKER to analyze or execute. For example, you may have a menu program, THEMENU.EXE, that is always called at the end of the AUTOEXEC.BAT to display a menu of programs and utilities. Obviously, it would be confusing if this menu program were repeatedly executed during the analysis phases of MEMMAKER.

You can use the file MEMMAKER.INF, in the DOS directory, to control which device drivers and programs MEMMAKER analyzes and executes. This file is ASCII and can be easily edited with EDIT.

Here is an example of MEMMAKER.INF:

```
;USED BY MEMMAKER
AUTOMENU
DBASE
DOSSHELL
EZMENU
LOTUS
MENU
NC
PARADOX
*PROTMAN
*SMARTDRV /DOUBLE_BUFFER
*SSWAP
*STRETCH
WORD
WP
WIN
123
```

All lines must be flush-left — that is, with no leading spaces. Any line beginning with a semicolon is ignored. The file contains a list of programs and device drivers, but the filename extension must not be specified; don't use EXE, COM, SYS, etc. Any program preceded by an asterisk is not modified by MEMMAKER but is executed. All other programs are completely ignored by MEMMAKER and are not run during the optimization process.

## Saving every drop of base memory

Most DOS and Windows users want to make base memory as large as possible. The first step is to use MEMMAKER to analyze your system and make the major changes, such as running HIMEM.SYS and EMM386.EXE, and loading programs high. MEMMAKER takes care of the big issues, but you can still squeeze out a few more bytes here and there.

NOTE

More and more programs and device drivers are being written in ways that conserve base memory and take direct advantage of UMBs or expanded memory. For example, the recent versions of the PCKWIK disk cache automatically loads into UMBs, and on my system, the fax board driver, SATISFAX.SYS, loads much of itself into expanded memory. In such cases, MEMMAKER does not try to directly load the program into UMBs. If you notice that MEMMAKER has not loaded a program high even though there is free upper memory, it is probably because the program loads high unassisted. You can always use the MEM command to verify where a program is being loaded and how much memory it is using.

TIP

Fine-tuning your system's memory configuration takes patience, as some changes can destabilize your system. In the event of a system lockup, you may need to retrace your steps and remove the "enhancements." Using these simple methods can help you keep track of your changes:

- If you are just making a change to one command's switches, don't make changes to the original line. Comment out the old line with the REM command and add a new statement with the appropriate switch changes. When I use the REM command, I always add the date so I can remember when I made the change. Some problems may not surface for weeks. Here is an example of how to adjust the EMM386 command:

```
REM 02/20/93 DEVICE=C:\DOS\EMM386.EXE RAM I=B000-B7FF NOEMS
DEVICE=C:\DOS\EMM386.EXE RAM I=B000-B7FF NOEMS MIN=0
```

- If you are making a few changes, make a backup copy of the configuration files before making the adjustments.
- Consider using the multiple configuration facility in DOS 6 (discussed in Chapter 8). Create a new menu option (for example, an option called *Experiment*) and save your old configuration in a separate configuration block.

The following section discusses some other changes that may gain you a few more bytes of base memory. I recommend that you don't make these changes until you are sure your system is running smoothly following MEMMAKER optimization. If you make too many changes at once, it is far more difficult to diagnose problems if they occur.

## Minimizing the environment size

The DOS environment is used to store the file search path, the DOS prompt format, the location of COMMAND.COM, and other information for specific programs (for example, a temporary directory for Windows). You can view your current environment settings by entering the SET command. Here is the environment on one of my systems:

```
CONFIG=NORMAL
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS;C:\;C:\WINDOWS;C:\WINDOWS\NDW;C:\BATFILES
PROMPT=$P$G
SAGEEDIT=C:\SPE
NU=C:\U\NORTON
TEMP=C:\TEMP
```

More and more programs are taking advantage of the environment to store information. The default size of the environment is 256 bytes, but that often is too small. You can use the SHELL command in CONFIG.SYS to increase the environment size by specifying the COMMAND switch /E. The following statement increases the environment size to 500 bytes:

```
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /E:500 /P
```

The space used by the environment is taken from base memory, so make sure that the environment isn't wasting space. Allocate only the environment space you need.

An easy way to determine the amount of environment bytes actually being used is to pipe the output of the SET command to a file. Here is an example:

```
c:\> set > sizeof.set
```

You can then use the DIR command to display the file size of the temporary file:

```
c:\> dir sizeof.set
```

The file size is a few bytes larger than the actual size of the environment data, but it provides a very close approximation.

Compare the consumed environment size with the memory allocated by the SHELL command. You may be able to save a few hundred bytes by reducing the amount allocated with the SHELL command. The environment space can range from 160 to 32,768 bytes and is always rounded up to the nearest number divisible by 16.

If you encounter the message `Out Of Environment Space`, you made the environment too small.

## Tuning the FILES statement

The FILES statement in CONFIG.SYS specifies the maximum number of files that DOS can have open at any one time (for example, FILES=25). The more files specified, the more memory is used. Every additional file above the default setting of 8 consumes approximately 50 bytes. You should therefore tune the FILES command to make sure that you are not allocating unnecessary memory for file handles. The wasted memory soon adds up; 20 files too many wastes 1,000 bytes.

There is no hard and fast way to accurately determine the number of files you need. It depends on the specific requirements of the software you use. If you run Windows, a common setting is FILES=50. If you don't run Windows but do run a database program, FILES=30 is reasonable. If you only use word processing or spreadsheet applications, FILES=20 might be enough.

If you encounter an "out of file handles" message or a similar message, you need to increase the number of files.

## Tuning the BUFFERS statement

The BUFFERS statement was originally introduced to speed up file operations. Think of it as a poor man's disk cache. You can allocate between 1 and 99 buffers; each buffer consumes approximately 500 bytes. If you load DOS high, the BUFFERS are loaded high as well.

Since its introduction, BUFFERS has been eclipsed by real disk caching software. If you use an 8086 system and don't own a disk cache, allocating about 20 buffers makes sense. Although it consumes a few thousand bytes of memory, it improves disk performance. However, if you own a 286 or later system, use DOS SMARTDRV or a third-party disk cache and reduce the buffers to 4. There is no advantage to having more than 4 buffers if a disk cache is used. In fact, having too many buffers can interfere with the disk cache and reduce the overall performance of your system.

## Optimizing the STACKS command

The STACKS command in CONFIG.SYS allocates memory to manage hardware interrupts. STACKS accepts two arguments: the number of stack frames and the size (in bytes) of each frame. Here is an example:

```
STACKS=9,256
```

This stacks statement consumes more than 2,000 bytes of base memory. Not all systems need to allocate memory for stacks, and so the memory is wasted. If no STACKS command is specified, DOS defaults to 9 stack frames of 128 bytes each.

Try using a STACKS=0,0 statement to remove all stack frame allocation. In most cases, your system will run fine. If you encounter stack error messages such as `Internal Stack Overflow`, you need to reinstate the STACKS command. Start with a low value, such as 9,128. If you still have a problem, increase the allocation to 9,256. If the problem persists, try 12,128, then 12,256, and so on, until the trouble is resolved.

### Minimizing the FCBS allocation

File control blocks are a legacy from the first version of DOS. Unless you are running some old (and probably very outdated) software, you do not need file control blocks. Conceptually, FILES is a modern version of FCBS.

When I removed the statement FCBS=20 from my CONFIG.SYS, I recovered an additional 944 bytes of memory. Even when there is no FCBS statement in CONFIG.SYS, however, DOS uses a default of FCBS=4. To minimize the FCBS allocation, add the statement `fcbs=1`. On my system, this produced a net saving of 1,120 bytes.

### Being frugal with LASTDRIVE

The LASTDRIVE command is used in CONFIG.SYS to allow for additional logical drives, such as network drives and RAM disks. For example, the statement LASTDRIVE=M instructs DOS to allocate enough memory for each drive up to the letter M. Because each drive consumes about 80 bytes, avoid allocating memory for drives that are never used. If you don't use RAM disks or networks, you might even omit the LASTDRIVE statement altogether.

### Minimize MSCDEX base memory consumption

MSCDEX, the CD-ROM driver, uses anywhere from 16K to 5K for sector buffers, depending on the configuration. If your system is configured for expanded memory, you can conserve base memory (or upper memory if MSCDEX is loaded high) by specifying the /E switch with MSCDEX. The /E switch instructs MSCDEX to place the CD-ROM buffers in expanded memory.

With default settings, MSCDEX will consume 35K of memory (high or base), but when /E is specified, the memory usage drops to 16K.

## Maximizing extended memory

The primary memory goal for users of DOS applications is to maximize the amount of free base memory. However, Windows users have a different goal: maximizing *extended* memory. In Real Mode, Windows uses expanded memory, but in Standard and Enhanced Modes, Windows thrives on extended memory.

In short, the most effective way to increase the amount of available extended memory is to install more memory. You can quickly ascertain how much extended memory is available by executing the MEM command. But don't run the command from within Windows because it will probably report that all the extended memory is in use (by Windows). If you find that no extended memory is available, make sure that the HIMEM.SYS and EMM386 statements are included in the CONFIG.SYS file.

Most of the previously discussed techniques for maximizing base memory do not adversely affect Windows performance. In fact, they improve the free memory available to applications when Windows launches a DOS program.

Some programs reserve extended memory for themselves, thereby preventing Windows from using that reserved memory. In the past, disk caches and expanded memory managers fell into this category, but both SMARTDRV and EMM386 can share extended memory cooperatively with Windows. For example, if Windows needs more extended memory to perform a certain operation, SMARTDRV automatically releases some of its extended memory to Windows. If you use a program that takes too much extended memory, such as a disk cache, refer to the program documentation to determine how to reduce the amount of extended memory it reserves.

If you run Windows on a system with minimal extended memory installed — say 2 to 4 megabytes — you may want to trade off the amount of free base memory for more extended memory, as follows:

- If you use EMM386 and you have not specified the NOEMS switch, EMM386 can convert extended memory to expanded. By default, EMM386 automatically converts some extended memory to expanded when it is loaded. If you specify the switch `min=0`, EMM386 conserves the extended memory and only converts it to expanded when a program requests it.
- If you use the NOEMS switch with EMM386, you can save additional extended memory by adding the NOVCPI switch. This removes support for VCPI compliant applications. As a rule of thumb, if you don't use EMS, you don't need VCPI support either.
- The HMA consumes 64K of extended memory, and the UMBs are created by mapping extended memory into upper memory. By disabling the EMM386 and DOS=HIGH commands from CONFIG.SYS, you free this memory for use by Windows. The amount of free conventional memory is likely to drop significantly because DOS, device drivers, and TSR programs are forced to reside in base memory. But, if you need extended memory, it's a trade-off worth considering.

## Managing expanded memory

The way EMM386 simulates expanded memory is significantly improved in DOS 6.

NOTE

If your computer uses expanded memory hardware, you must use a third-party expanded memory device driver, such as EMM.SYS. DOS 6 does not provide direct expanded memory managers — only ways to simulate expanded memory with extended.

With earlier versions, you could instruct EMM386 to reserve part of extended memory to emulate expanded memory using the MEMORY switch. The main weakness of this approach was that the memory was literally reserved and could only be used for expanded memory.

DOS 6's EMM386.EXE is a significant improvement over its predecessor because it allows a 386 or later computer to share memory from a *pool.* If a program needs expanded memory, memory is taken from the pool and presented to the program as expanded memory. Similarly, if a program requires extended memory, it too can be allocated from the common memory pool.

Provided that EMM386 is specified without the NOEMS switch, expanded memory is directly supported.

The EMM386 MEMORY switch sets the maximum amount of expanded memory that is provided, and the MIN= switch sets the minimum. Five other switches also are used to manage the EMS page frame: MX, FRAME=, /P, PN, and H=. For further information on these switches, refer to Part IV, "The Complete DOS 6 Command Reference."

## Solving memory conflicts

"Lockup happens" — what a perfect bumper sticker. When juggling software configurations and program-loading sequences, you will eventually run into a situation where the system stops behaving. A program refuses to run, or worse still, the system won't respond. My favorite techno-slang for this situation is *hosed,* but *locked up* and *hung* are also popular.

The following paragraphs describe some common ailments and suggested remedies.

### Running MEMMAKER locks up the system

First, reboot the system. MEMMAKER normally recognizes that there has been a problem and restores your original configuration files. If MEMMAKER locks up second time, reboot again and press F5 when you see the message `Starting MS-DOS...` to *clean boot.* Change to the DOS directory and execute the following command:

```
memmaker /undo
```

MEMMAKER then restores your old configuration files and reboots your system. Try running MEMMAKER again, but this time choose the custom option. Then change the Scan Upper Memory Aggressively? option to No and try again.

### The new configuration locks up the system

If the system is locked at the DOS prompt or repeatedly hangs after a few simple commands, there probably is a memory conflict. You need to do some detective work to isolate the true cause of the problem.

If the system refuses to run and you recently changed CONFIG.SYS or AUTOEXEC.BAT, it is likely that one of the recent changes caused the problem. To identify the specific cause, try reverting back to the previous configuration and making the changes one by one.

If you don't know which changes were made recently, take a methodical approach to isolate the problem. One tactic is to strip the CONFIG.SYS and AUTOEXEC.BAT files down to the bare bones and start including statements one by one until the problems start occurring. If you like to play a hunch, you might go in the opposite direction and remove individual statements until the problem goes away. Remember that you can use the F8 key at start-up to interactively select which statements execute. Often, problems are caused by a conflict in the UMBs. Rather than remove device drivers altogether, you might change the DEVICEHIGH statements to `device`.

If you are unable to isolate the specific program that is causing the trouble, EMM386 may be to blame. By adjusting the EMM386 switches, you should be able to isolate the problem.

Start by removing the HIGHSCAN switch if it is present. HIGHSCAN instructs EMM386 to use an aggressive technique to identify every free UMB space. This can cause problems on some systems, especially those that use SCSI device drivers, like ASPI4DOS.

GEM

Many ISA (industry standard architecture) bus systems equipped with SCSI drives cannot handle more than 16MB of memory. The problems arise because the DMA (direct memory access), used by SCSI device drivers, on ISA systems can only access 16MB of RAM. Unfortunately, the common solution is to change the CMOS to recognize only the first 16MB of installed memory, thereby disabling the additional installed memory. Refer to Chapter 7 for a thorough discussion of ISA and DMA.

If HIGHSCAN wasn't the problem, try adding the NOMOVEXBDA switch to stop EMM386 from relocating the extended BIOS data area. If that doesn't work, remove the switch, along with any include (I=) statements. If it does work, isolate the problem memory area by adjusting the I= statements until the problem returns.

Lockup may also be caused by EMM386 mapping extended memory into an active area of upper memory. A quick way to isolate this kind of problem is to exclude all upper memory with the statement X=A000-FFFF. If the problem appears to be fixed, try excluding a smaller area — for example, use X=A000-DFFF. Keep whittling the memory region down until you isolate one specific region of memory. You can use the resulting X switch to stop EMM386 from using that area of upper memory.

### A program causes system lockup

Some programs do not like being loaded into upper memory. If a program locks the system when it is executed, try loading it in base memory by removing the LOADHIGH (or LH) statement.

If the program wasn't being loaded high in the first place, you may need to isolate the conflict in the ways described in the last section. In other words, start from a simple configuration that works with the program and keep expanding the statements until the program locks up again.

You should, of course, refer to the program documentation to determine whether there are known conflicts. It is also worth calling the vendor's technical support line to ask for guidance.

### A program runs more slowly

This problem might occur when the NOEMS switch is added to EMM386. If that is the case, the program probably uses expanded memory to speed operations. Remove the NOEMS switch and see whether the speed improves.

If the speed doesn't improve, check to see whether the disk cache has been disabled or reduced in size. Also, make sure that you don't have two disk caches loaded because this too slows the system down.

Once again, if you can't seem to find the cause of the slowdown, start with a bare-bones configuration and add the statements one by one. You should be able to isolate the cause of the problem.

### The system doesn't respond to Ctrl-Alt-Del

Add the ALTBOOT switch to the EMM386 statement in CONFIG.SYS if your system doesn't reboot when you press Ctrl-Alt-Del or if your system responds to the keystrokes in an erratic way.

## Summary

This chapter explored PC memory, one of the most important topics for today's DOS user. You learned some of the following points:

- The most common PC memory chip is the DRAM chip. High-capacity chips are soldered onto compact circuit boards known as SIMMs. High-performance SRAM chips are used for memory caches, but their power demands are too high for general-purpose RAM.
- Each memory location has a segment:offset address. On 286 and later systems, the segment is used in a lookup table to determine the physical address. This technique increases the maximum addressable memory to 16MB on a 286 and all the way to 4 gigabytes on 386+ systems. (On 8086 systems, the maximum addressable memory is 1MB.)
- The first megabyte of memory is separated into base memory and upper memory. Additional memory is either expanded or extended memory.
- On 286+ systems, you can use HIMEM.SYS to remap the first 64K of extended memory as *high* memory. The majority of DOS can be loaded into high memory.
- On 386+ systems, unused areas of upper memory can be filled with extended memory, creating UMBs.
- MEMMAKER analyzes your system and modifies CONFIG.SYS and AUTOEXEC.BAT to make optimum use of your system's memory.
- You cure memory lockup by methodically revising the configuration files until you isolate the cause of the problem.

Chapter 6 shows how you can use DOS device drivers to brighten the display and customize the keyboard.

## Chapter 6

# Taking Control of the Display and Keyboard

---

### In This Chapter

- A more colorful way to look at DOS
- ANSI.SYS techniques
- Ways to jazz up the screen display
- How to change display lines and modes
- How to customize the keyboard
- Alternatives to ANSI.SYS

---

## We Want a Color TV! We Want a Color TV!

In the 60s, my sisters and I put our father under a lot of pressure. Every night, we begged him to buy a color TV. Eventually, we got one. It appears that, unlike my sisters and I, the DOS developers had a monochrome upbringing. Let's face it — DOS is not very pretty.

If you have been wanting a colorful version of DOS, you might be surprised to hear that you already have one. By using the DOS device driver ANSI.SYS, you can add a lot of pizazz to your DOS sessions. As well as offering color support, ANSI.SYS provides ways to reposition the cursor and customize the keyboard. You can assign keystroke sequences to the function keys, for example. ANSI.SYS is powerful, yet it takes less than 5K of memory.

## Reviewing ANSI.SYS Techniques

To take advantage of ANSI.SYS, you must install it as a device driver in CONFIG.SYS. You can load it using either DEVICE or DEVICEHIGH.

ANSI.SYS supports two switches, both of which are related to extended (101-key) keyboards. On extended keyboards, you can use more than one key to perform some operations. For example, two keys enter the plus (+) character, and two keys perform the Page Up function. The /X switch instructs ANSI.SYS to treat these keys independently. With this switch in effect, you can remap the keys so that each has a different purpose.

If you have a nonstandard extended keyboard and experience problems remapping keys, use the /K switch to instruct ANSI.SYS to treat the keyboard as a traditional 84-key keyboard. You cannot use the /X and /K switches together.

**NOTE**

If you specified /K in a SWITCHES entry in CONFIG.SYS, load ANSI.SYS with the /K switch. If you use /K with SWITCHES, you must also use it with ANSI.SYS.

The following command from CONFIG.SYS loads ANSI.SYS into upper memory and specifies individual key support on extended keyboards:

```
devicehigh=c:\dos\ansi.sys /x
```

## Sending instructions to ANSI.SYS

Whether you want to change display colors, position the cursor, or remap keys, you need to communicate with ANSI.SYS. You give instructions to ANSI.SYS by sending special escape codes to the standard output device (that is, the display).

The escape codes all begin with two specific *signature characters:* the Esc character, which has the ASCII value 027, and the left square-bracket character ( [ ), which has the ASCII value 019. The characters you place after the signature characters depend on the task you want to perform; the character sequence 33m, for example, sets the text foreground color to brown.

ANSI.SYS intercepts characters sent to the standard output device (the display) but does nothing unless it receives the signature characters. When it receives the signature characters, ANSI.SYS jumps into action and responds to the instructions that follow.

The DOS commands ECHO, PROMPT, and TYPE all send characters to the standard output device, so you can use them to send escape sequences to ANSI.SYS. You need to be aware of some conditions, however.

Suppose that you want to change the foreground color to brown. You might think that the following command would do the job:

```
echo Esc[33m
```

NOTE

In the examples, `Esc` represents the Esc key (ASCII code 027); don't enter the literal characters *E, s,* and *c.*

It isn't quite that straightforward. When you press Esc, DOS thinks that you want to undo the partially entered command, and so it clears the command before you have a chance to enter the [33m characters. In many situations, holding down the Alt key and pressing the 3-digit ASCII value on the numeric keypad provides an alternate way of entering a special key. If you use this technique on the command line, however, DOS reacts as if you pressed Esc directly — it clears the command. Shucks.

How do you solve this problem? Read on.

## Using batch files

One way to get around the problem described in the preceding paragraphs is to create a batch file that contains the ECHO statement(s). The following batch file, for example, changes the text color to brown:

```
@echo off
echo ←[33m
```

GEM

This approach presents another challenge, however. When you press the Esc key in EDIT, nothing happens. The same goes for entering the Alt three-key sequence. You can use a simple trick to solve the problem. In EDIT, you can enter a special character by pressing Ctrl-P and then holding down the Alt key and entering the two or three digits identifying the character's ASCII value. Entering the Esc key is even easier: Just press Ctrl-P and then press the Esc key.

Note that EDIT displays the Esc character as a left-pointing arrow (←).

After creating a batch file containing the ANSI.SYS escape sequences, just execute the batch file, and ANSI.SYS responds. If you issue the command just discussed in the example, the foreground color changes to brown.

DISK

More often than not, you want to send more than one escape sequence to ANSI.SYS. You might want to change the foreground and background colors and clear the screen, for example. The following batch file, ANSICODE.BAT, makes it easy to send one or more commands to ANSI.SYS:

```
@echo off
:LOOP
if "%1" == "" goto quit
```

```
echo ←[%1
shift
goto loop
:QUIT
```

If you use this neat little batch file, you don't have to worry about entering the Esc[ character sequence because it is already embedded in the batch file. To change the foreground color to brown, all you need to do is enter the following command:

```
ansicode 33m
```

Better still, ANSICODE.BAT can send several ANSI codes in one command. The following command, for example, changes the foreground to brown and the background to green and clears the screen:

```
ansicode 33m 42m 2J
```

One of the special ANSI.SYS escape sequences is Esc[2J, which clears the screen and moves the cursor to the home position. (Also, the Esc[K command erases the line starting at the cursor position.)

For more information about batch files, refer to Chapter 10.

## Using the TYPE command

Instead of using the ECHO statement in batch files, you can just enter the commands into a plain file and then use TYPE to write the commands to the display. For example, you can use EDIT to create a single-line text file, BROWN.COL, that contains the following characters:

```
←[33m
```

Then, by entering the following command, you can direct the escape sequence to ANSI.SYS:

```
type brown.col
```

## Using PROMPT

Like EDIT and TYPE, PROMPT sends characters to the display, so you can use it to communicate with ANSI.SYS. One of the special meta-characters supported by PROMPT is $e, which represents the Esc character.

For a complete list of all the meta-characters supported by PROMPT, refer to the PROMPT entry in Part IV, "The Complete DOS 6 Command Reference."

Using the same example discussed earlier, you could change the text foreground color to brown by entering the following command:

```
prompt $e[33m
```

Although ANSI.SYS correctly intercepts the commands, this approach does have an irritating drawback: It changes the system prompt, usually to a null (or characterless) prompt. After using PROMPT to send ANSI.SYS one or more instructions, call PROMPT again to reset the DOS prompt. The following batch file, ANSIPRMT.BAT, makes life a little easier. It saves the old prompt in a batch file and then restores it after the ANSI.SYS instructions are processed:

```
@echo off
set|find "PROMPT" > oldprmpt.bat
:LOOP
if "%1"=="" goto stop
prompt $e[%1
@echo on
echo.
@echo off
shift
goto loop
:STOP
call oldprmpt
del oldprmpt.bat
```

Again, don't worry if some of the batch file statements seem like Greek to you. All will be revealed in Chapter 10.

If you know the system's default prompt, you can simplify matters by adding the default prompt settings to the end of the ANSI.SYS commands. For example, if you normally use the $p$g prompt, which displays the current directory as the prompt, you can set the color to brown and reset the prompt to normal in one command, as follows:

```
prompt $e[33m$p$g
```

## Sending multiple escape sequences

You do not need to confine each ANSI.SYS instruction to a single line. You can send multiple instructions on a single line by preceding each new instruction with the standard Esc[ sequence.

After changing the foreground color to brown, you can clear the screen by using the 2J command as follows:

```
@echo off
echo ←[33m
echo ←[2J
```

Alternatively, you can combine the two commands into a single statement:

```
@echo off
echo ←[33m←[2J
```

The letters used in ANSI.SYS commands are case sensitive. If you enter the 2J sequence as 2j, the screen does not clear.

## Determining whether ANSI.SYS is loaded

If ANSI.SYS is not installed, DOS does not react very gracefully to all the obscure escape sequences. Although nothing untoward happens, the escape code gibberish is written to the screen and looks very unprofessional. A sophisticated display screen that relies on ANSI.SYS to change colors and attributes, position the cursor, and redefine keys looks dreadful if ANSI.SYS is not installed. If you write a batch file that uses ANSI.SYS features, you must check to see whether ANSI.SYS is loaded.

With earlier versions of DOS, you had to write a program, using DEBUG or another programming language, to determine whether ANSI.SYS was installed in memory. Thanks to the introduction of the MEM /M switch, this is no longer necessary. DOS displays the details of where ANSI.SYS is installed in memory if you enter the following command:

```
mem /m ansi
```

If ANSI.SYS is not installed, the program responds with the following message:

```
ansi is not currently in memory.
```

You can use FIND to search for the output "not currently" and then test FIND's ERRORLEVEL to determine whether ANSI.SYS is or is not installed.

The following batch file, ANSIOK.BAT, displays a message that indicates whether ANSI.SYS is installed:

```
@echo off
mem /m ansi|find "not currently" /i > nul
if errorlevel 1 goto loaded
:NOTLOADED
echo ANSI.SYS is not in memory
goto quit
:LOADED
echo ANSI.SYS is loaded
rem Insert your ANSI statements here

:QUIT
```

You also can employ the technique of using MEM /M together with FIND to test whether any specific program, such as Share, Windows, or DOSKEY, is loaded in memory. The approach is explored in detail in Chapter 10.

You can improve the ANSICODE.BAT file discussed earlier by adding a test to determine whether ANSI.SYS is loaded, as follows:

```
@echo off
mem /m ansi|find "not currently" /i > nul
if errorlevel 1 goto loaded
:NOTLOADED
echo ANSI.SYS is not in memory
goto quit
:LOADED
:LOOP
if "%1" == "" goto quit
echo ←[%1
shift
goto loop
:QUIT
```

## Jazzing Up the Display

You just learned a half-dozen ways to set the text color to brown. You will, no doubt, be pleased to hear that you can do a lot more. For example, you can set the foreground and background colors of the text, write any characters in any

### The History of ANSI Commands

You might be wondering who dreamed up the awkward notion of sending commands to the screen by using obscure character sequences that can't even be typed on the command line. It was the American National Standards Institute (ANSI). In fairness to ANSI, however, I should explain that these commands were standardized long before DOS came on the scene.

The commands were drawn up primarily for the benefit of mini- and mainframe computers. When CRTs (monitors) were first implemented on the larger computers, they basically behaved like paper-fed terminals. The typical workstation was just a CRT and a keyboard, with no independent computational capability. The primary role of these workstations was to display computer prompts and send keyboard input to the main computer. Such workstations were often called *dumb terminals*. Everything was drawn on a line-by-line basis, and old commands just scrolled off the top of the screen — no support was provided for full-screen editing or data entry. The ANSI standards defined a command set that allowed the CRT to behave like a full-screen terminal rather than a line-oriented Teletype. I, for one, was excited at the time.

The latest ANSI command set was ratified in 1979 and is called *ANSI standard X3.64*. Similar standards were implemented for printers in order to standardize such commands as line feed and form feed.

The ANSI.SYS device driver implements the screen-related commands that were originally developed for the mini- and mainframe dumb terminals.

position on the screen, and change the display attribute of any text to bold, blink, and reverse video.

## Setting foreground and background colors

You can set the foreground and background display colors using the m command set. The syntax of the command is as follows:

Esc[*code*m

You replace the variable *code* with a two-digit number identifying the desired color. Table 6-1 lists all the valid color codes.

The following batch file sets the display color to cyan text on a blue background:

```
@echo off
echo ←[36m
echo ←[44m
```

If you are entering multiple m commands, you can issue them in one instruction by using the following syntax:

Esc[*code*;*code*; ...m

For example, the following line from a batch file sets the display colors to black text on a green background:

```
echo ←[30;42m
```

**Table 6-1 ANSI.SYS Display Codes**

| *Color* | *Foreground Code* | *Background Code* |
|---|---|---|
| Black | 30 | 40 |
| Red | 31 | 41 |
| Green | 32 | 42 |
| Brown | 33 | 43 |
| Blue | 34 | 44 |
| Magenta | 35 | 45 |
| Cyan | 36 | 46 |
| Light gray | 37 | 47 |

## Setting display attributes

The ANSI.SYS m command provides a set of single-digit codes for changing the text display attribute. Table 6-2 lists the valid codes.

The most commonly used attribute code is 1, which changes the dull default colors into bright high-intensity colors; brown becomes yellow, light gray becomes bright white, etc.

When executed, the following batch file, ANSIATTR.BAT, illustrates each of the attribute effects:

```
@echo off
echo ←[2J
echo ←[0m All attributes off
echo ←[0;1m Bright intensity
echo ←[0;4m Underline (monochrome systems only)
echo ←[0;5m Blinking text
echo ←[0;7m Reverse Video
echo ←[0;8m Hidden or invisible
echo ←[1;33m
```

The attribute 8m hides the text and the cursor. If this is the last command in the batch file, the system prompt and the cursor are invisible after the batch file finishes. This is a useful trick in some instances — for example, when you want to suppress the display of program or batch file activity. The preceding ANSIATTR.BAT batch file, however, sets the text foreground color to yellow (bright brown) before terminating.

**Table 6-2 ANSI.SYS Display Attribute Codes**

| *Code* | *Attribute* |
|---|---|
| 0 | All attributes off |
| 1 | Bright intensity |
| 4 | Underline (monochrome systems only) |
| 5 | Blinking text |
| 7 | Reverse video |
| 8 | Hidden or invisible |

## Positioning the cursor

Every PC programming language provides ways to write text in any color at any location on the screen. Normally, DOS positions the cursor at the line following an ECHO statement, making it impossible to write text on lines above the cursor without clearing the screen first. Impossible, that is, unless ANSI.SYS is installed.

The ANSI.SYS H commands support the positioning of the cursor anywhere on the display. After you position the cursor, any subsequent text you enter with the ECHO command is written to the display starting at the cursor location.

The syntax of the H command is as follows:

```
Esc[row;colH
```

The *row* variable is the row or line number, starting with 1 at the top of the screen and ending with 25, 43, or 50 at the bottom. The *col* parameter is the column or X coordinate, starting with 1 at the left and ending with 40 or 80 at the right.

The following batch file writes a string to the top right corner of the display:

```
@echo off
echo ←[1;69HPretty Neat
```

The only problem with this batch file is that the cursor remains at the top of the display. After the batch file finishes, the DOS prompt rests at the top of the display, in the middle of old commands. Very ugly.

Two more ANSI.SYS commands can resolve this problem. The s command saves the current cursor position, and the u command moves the cursor back to the position it held when the last s command was executed.

The following batch file, WRITEAT.BAT, provides a generic routine for writing text at any location on the screen and then returns the cursor to its prior location:

```
@echo off
if "%1"=="" goto quit
echo ←[s
echo ←[%1;%2H%3 %4 %5 %6 %7 %8 %9
echo ←[u
:QUIT
```

In addition to the position, save, and restore commands, ANSI.SYS also supports column- and row-based cursor movement commands relative to the active cursor position. Table 6-3 summarizes all the cursor-related commands supported by ANSI.SYS.

**Table 6-3 ANSI.SYS Cursor Control Codes**

| *Escape Sequence* | *Description* | *Action* |
|---|---|---|
| Esc[*row*;*col*H | Absolute | Moves the cursor to the specific row and column. If only one value is specified (without a semicolon), it is assumed to represent the column, and the row defaults to 1. |
| Esc[s | Save | Saves the active cursor position. |
| Esc[u | Restore | Repositions the cursor to the position it occupied when the last s (save) command was issued. |
| Esc[*pos*A | Up | Moves the cursor up the number of rows specified by the *pos* variable. The cursor stops when it reaches the first row. |
| Esc[*pos*B | Down | Moves the cursor down *pos* rows. The cursor stops when it reaches the last row. |
| Esc[*pos*C | Right | Moves the cursor to the right *pos* columns. The cursor stops when it reaches the last column. |
| Esc[*pos*D | Left | Moves the cursor left *pos* columns. The cursor stops when it reaches the first column. |

***Note:*** The f command has the same meaning and syntax as the H command for moving the cursor to a specific address.

## Pulling it all together

By giving you the ability to change colors and attributes, position the cursor, and write text, ANSI.SYS provides all the tools you need to create a very snazzy and professional display. These displays are well suited to batch files for help systems, menus, and the like.

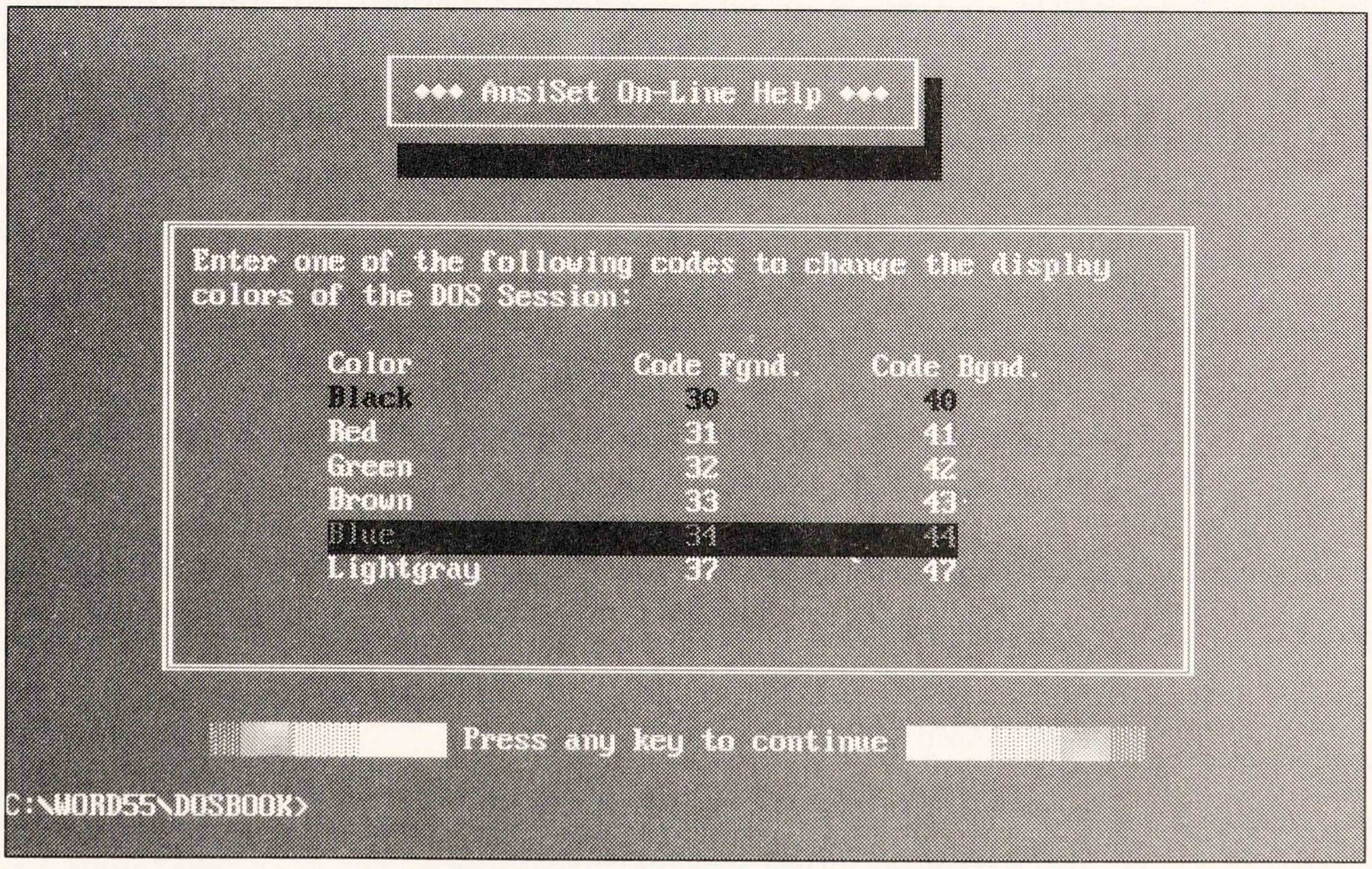
The following statements are extracted from the batch file, ANSISET.BAT, which is a derivative of ANSICODE.BAT, discussed earlier:

```
@echo off
if “%1”==”?” goto help
if “%1”==”/?” goto help
mem /m ansi | find “not currently” /i > nul
if errorlevel 1 goto loaded
:NOTLOADED
echo ANSI.SYS is not in memory
goto quit
```

```
:LOADED
:LOOP
if "%1" == "" goto quit
echo ←[%1m
shift
goto loop
:HELP
rem Clear the screen with a blue background and draw banner
echo ←[1;37;44m←[2J
echo ←[2;23H ┌──────────────────────────────┐
echo ←[3;23H │←[3;54H│
echo ←[4;23H └──────────────────────────────┘
echo ←[3;29HAnsiSet On-Line Help
echo ←[3;25H←[0;35;44m♦←[37;44m♦←[36;44m♦
echo ←[3;50H←[0;36;44m♦←[37;44m♦←[35;44m♦
rem Draw Box shadow
echo ←[5;24H←[0;30;44m████████████████████████████████
echo ←[3;55H█←[4;55H█
echo ←[1;37;44m
echo ←[7;10H╔═══════════════════════════════════════════════════════════╗
echo ←[8;10H║←[59C∫
echo ←[9;10H║←[59C∫
echo ←[10;10H║←[59C∫
echo ←[11;10H║←[59C∫
echo ←[12;10H║←[59C∫
echo ←[13;10H║←[59C∫
echo ←[14;10H║←[59C∫
echo ←[15;10H║←[59C∫
echo ←[16;10H║←[59C∫
echo ←[17;10H║←[59C∫
echo ←[18;10H║←[59C∫
echo ←[19;10H║←[59C∫
echo ←[20;10H╚═══════════════════════════════════════════════════════════╝
echo ←[1;33;44m←[8;12HEnter one of the following codes to change the display
echo ←[9;12Hcolors of the DOS Session:
echo ←[11;20H←[1;37;44mColor     Code Fgnd.  Code Bgnd.
echo ←[12;20H←[0;30;44mBlack        30          40
echo ←[13;20H←[0;31;44mRed          31          41
echo ←[14;20H←[0;32;44mGreen        32          42
echo ←[15;20H←[0;33;44mBrown        33          43
echo ←[16;20H←[0;34;40mBlue       34          44
echo ←[17;20H←[0;35;44mMagenta    35          45
echo ←[17;20H←[0;36;44mCyan       36          46
echo ←[17;20H←[0;37;44mLightgray  37          47
echo ←[22;13H←[1;37;44m░▒▓█
echo ←[22;54H←[1;37;44m█▓▒░
echo ←[22;28H←[5;36;44mPress any key to continue
rem Wait for a key
pause > nul
rem Position cursor at bottom of screen, set colors, and quit
echo ←[23;1H←[0;1;37;44m
:QUIT
```

The code creates a fancy help screen that is displayed if the user executes the batch file with the /? or ? switch. Figure 6-1 shows the display that is generated when this batch file is executed.

The preceding batch file, along with all the others mentioned in the book, are included on the Bob's Shareware Collection disks. Run it for yourself to see the kind of colorful effects you can create.



```
            ♦♦♦ AnsiSet On-Line Help ♦♦♦

      Enter one of the following codes to change the display
      colors of the DOS Session:

           Color                  Code Fgnd.   Code Bgnd.
           Black                     30           40
           Red                       31           41
           Green                     32           42
           Brown                     33           43
           Blue                      34           44
           Lightgray                 37           47

               Press any key to continue

C:\WORD55\DOSBOOK>
```

**Figure 6-1:** The help screen that appears when ANSISET.BAT is executed with the /? switch.

## Creating fancy prompts

You can use ANSI.SYS escape codes to colorize your system prompt. As a Houston resident, I consider it my unwritten duty to use the infamous Texas Lone Star flag as an example.

The following prompt statement (located in TEXAS.BAT) is mandatory in Texas:

```
@echo off
PROMPT $e←[0;44m*$e←[41m|$e←[40;36m $p$g $e←[1;33;40m
```

The Californian bear prompt is not quite as easy!

You Bill the Cat lovers will enjoy the following prompt (located in BILLCAT.BAT):

```
@echo off
prompt $e[6C/$b$_    \'o.0'$_
$q(___)$q$_$e[1;5;31mU$e[0;37m$_ACK!PFFTTT!$_$P$G
```

# Changing Display Lines and Modes

More and more PCs are using EGA and VGA display systems. Because these EGA and VGA systems support a condensed display, capable of displaying 43 or 50 lines of text, respectively, Microsoft enhanced ANSI.SYS (starting with DOS 5) to support them.

To set your PC to run DOS in Condensed Mode, you execute the MODE command (ANSI.SYS must be installed). The syntax of the MODE command for changing the display lines is as follows:

```
mode con: lines=xx
```

The *xx* parameter represents the desired number of display lines and can have the value 25, 43, or, on VGA systems, 50.

The following batch file, SET43.BAT, sets the display to 43-line mode:

```
@echo off
mem /m ansi|find "not currently" /i > nul
if errorlevel 1 goto loaded
:NOTLOADED
echo ANSI.SYS is not in memory
goto quit
:LOADED
mode con: lines=43
:QUIT
```

You also can use the MODE command to set the number of characters per line to 40 or 80. Refer to Part IV, "The Complete DOS 6 Command Reference," for further details.

You can use the ANSI.SYS =h command to set any of the display modes listed in Table 6-4.

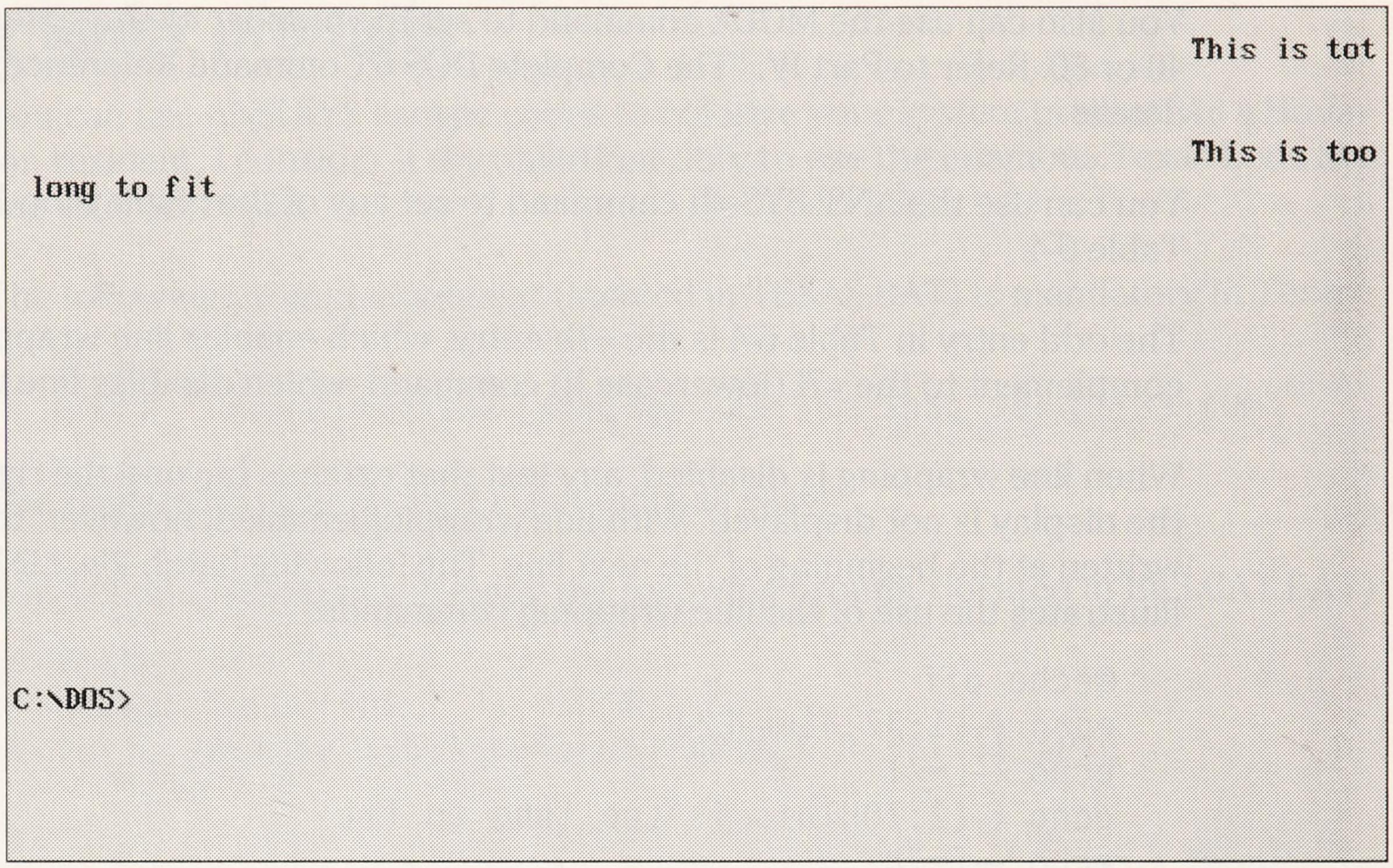
The odd entry in Table 6-4 is the =7h entry, which enables line wrapping. It is the complement to the =7l (lowercase L) command, which disables line wrapping.

When line wrapping is disabled, any text that extends beyond the right edge of the display is not displayed. With line wrapping enabled, however, the text is written at the beginning of the next line. The following batch file, LINEWRAP.BAT, illustrates the use of the line wrapping commands:

```
@echo off
echo ←[1;37;41m←[2J
echo ←[=7l
echo ←[2;70HThis is too long to fit
echo ←[=7h
echo ←[5;70HThis is too long to fit
echo ←[20;1H
```

**Table 6-4 ANSI.SYS Display Mode Control Codes**

| *Escape Sequence* | *Description* |
|---|---|
| `Esc[=0h` | Text — monochrome, 40 columns by 25 lines |
| `Esc[=1h` | Text — color, 40 columns by 25 lines |
| `Esc[=2h` | Text — monochrome, 80 columns by 25 lines |
| `Esc[=3h` | Text — color, 80 columns by 25 lines |
| `Esc[=4h` | Graphics — 4-color, 320 by 200 pixels |
| `Esc[=5h` | Graphics — monochrome, 320 by 200 pixels |
| `Esc[=6h` | Graphics — monochrome, 640 by 200 pixels |
| `Esc[=7h` | Enables line wrapping |
| `Esc[=13h` | Graphics — color, 320 by 200 pixels |
| `Esc[=14h` | Graphics — 16-color, 640 by 200 pixels |
| `Esc[=15h` | Graphics — monochrome, 640 by 350 pixels |
| `Esc[=16h` | Graphics — 16-color, 640 by 350 pixels |
| `Esc[=17h` | Graphics — monochrome, 640 by 480 pixels |
| `Esc[=18h` | Graphics — 16-color, 640 by 480 pixels |
| `Esc[=19h` | Graphics — 256-color, 320 by 200 pixels |



```
                                                                     This is tot

                                                                     This is too
long to fit






C:\DOS>
```

**Figure 6-2:** ANSI.SYS allows line wrapping to be turned on and off. The top right line was written with wrapping off and is truncated. The line below was written with wrapping on, and so the text spills over to the beginning of the next line.

Figure 6-2 shows the display generated by the batch file.

Disabled line wrapping has an eccentric feature: Characters that extend beyond the right of the screen are all written (in turn) to the rightmost character position. Notice in Figure 6-2 that the first line displays the string `This is tot`. The last character, *t,* is actually the character from the end of the original string — it's the *t* from *fit.*

## Customizing the Keyboard

One of the most valuable features of ANSI.SYS is its capacity to remap keys and assign commands (or strings) to keys. For example, using ANSI.SYS, you can reverse the arrangement of the semicolon and colon keys, assign the HELP command to the F1 key, assign the EDIT command to F2, and so on.

### Key codes

The key (pun intended) to using ANSI.SYS to customize the keyboard is knowing the scan codes for each key. Each keystroke you type is identified by a one- or two-digit *scan code.* The uppercase J has a single-byte scan code value of 74, for instance, and the F10 key has a two-byte scan code of 0 and 69. Most of the special or nonalphanumeric keys are assigned two-byte scan codes, with the first byte having a value of 0.

## Understanding the Mechanics of Scan Codes

The keyboard unit includes a keyboard controller, which is responsible for monitoring and reporting keyboard activity. When you press a key on the keyboard, two events are reported: the key being pressed and the key being released. Each key is assigned two codes, with one code representing the key press and the other representing the key release. These single-byte codes are sometimes referred to as the *make code* and the *break code,* respectively. The make code for the J key is 24h, and the break code is A4h.

Although most keys generate single-byte codes, some of the new extended keys generate double-byte codes. A code of E0h indicates that an extended key was pressed, and a second code indicates the specific key. For example, the gray PgUp key on the extended keypad is identified by the byte pair E0h and 49h; the gray Home key has the byte pair E0h and 47h. Each physical key on the keyboard has a unique code.

When you press or release a key, the keyboard controller generates interrupt 9h, which advises the microprocessor that keyboard activity is taking place. The interrupt handler for interrupt 9h is in ROM-BIOS. This interrupt handler interprets the stream of bytes and then converts the codes into two-byte scan codes, which identify the key.

Before making the conversion, the interrupt handler must check the status of the three shift keys (Ctrl, Alt, and Shift) and the toggle keys (Caps Lock and Num Lock). The status of these special keys is recorded in ROM-BIOS at location 0040:0017. This enables the system to assign different scan codes based on the status of the shift and toggle keys. For example, the interrupt handler might receive two separate codes, 24h followed by A4h, indicating that the J key was pressed and released. The interrupt handler then checks the status of the shift keys and the Caps Lock key to determine whether J, j, Ctrl-J, or Alt-J should be returned as the scan code.

In Chapter 11, you learn how to write compact utilities for checking and setting the data at memory location 0040:0017. Using these utilities, you can set the value of the Caps Lock and Num Lock keys. In a batch file, you can test whether the Alt key is pressed and, if so, skip past confirmation prompts, suppress the display of messages, or perform other useful tasks.

After making the translation, the ROM-BIOS interrupt handler places the scan codes into the keyboard buffer, which is located at memory address 0040:001E. Programs (including DOS) access this buffer to determine user input.

During the translation process, the interrupt handler looks for one of the following four special key combinations: Ctrl-Alt-Del, Shift-PrtScr, Ctrl-Num Lock, and Ctrl-Break. When one of these key sequences occurs, ROM-BIOS immediately invokes an interrupt to alert the microprocessor instead of placing the keystrokes into the keyboard buffer.

All that for a simple key press. *Phew!*

***Note:*** The subject of interrupts and interrupt handlers is discussed in detail in the next chapter.

When using ANSI.SYS to manage the keyboard, you must specify the key you want to redefine or modify. When you specify two-byte codes, separate each byte with a semicolon. Table 6-5 details all the scan codes supported by ANSI.SYS.

**Table 6-5 ANSI.SYS Keyboard Control Codes**

| *Key* | *Standard* | *With Shift* | *With Ctrl* | *With Alt* |
|---|---|---|---|---|
| A | 97 | 65 | 1 | 0;30 |
| B | 98 | 66 | 2 | 0;48 |
| C | 99 | 67 | 3 | 0;46 |
| D | 100 | 68 | 4 | 0;32 |
| E | 101 | 69 | 5 | 0;18 |
| F | 102 | 70 | 6 | 0;33 |
| G | 103 | 71 | 7 | 0;34 |
| H | 104 | 72 | 8 | 0;35 |
| I | 105 | 73 | 9 | 0;23 |
| J | 106 | 74 | 10 | 0;36 |
| K | 107 | 75 | 11 | 0;37 |
| L | 108 | 76 | 12 | 0;38 |
| M | 109 | 77 | 13 | 0;50 |
| N | 110 | 78 | 14 | 0;49 |
| O | 111 | 79 | 15 | 0;24 |
| P | 112 | 80 | 16 | 0;25 |
| Q | 113 | 81 | 17 | 0;16 |
| R | 114 | 82 | 18 | 0;19 |
| S | 115 | 83 | 19 | 0;31 |
| T | 116 | 84 | 20 | 0;20 |
| U | 117 | 85 | 21 | 0;22 |
| V | 118 | 86 | 22 | 0;47 |
| W | 119 | 87 | 23 | 0;17 |
| X | 120 | 88 | 24 | 0;45 |
| Y | 121 | 89 | 25 | 0;21 |
| Z | 122 | 90 | 26 | 0;44 |
| 1 | 49 | 33 | N/A | 0;120 |
| 2 | 50 | 64 | 0 | 0;121 |

*(continued)*

### Table 6-5 ANSI.SYS Keyboard Control Codes *(continued)*

| *Key* | *Standard* | *With Shift* | *With Ctrl* | *With Alt* |
|---|---|---|---|---|
| 3 | 51 | 35 | N/A | 0;122 |
| 4 | 52 | 36 | N/A | 0;123 |
| 5 | 53 | 37 | N/A | 0;124 |
| 6 | 54 | 94 | 30 | 0;125 |
| 7 | 55 | 38 | N/A | 0;126 |
| 8 | 56 | 42 | N/A | 0;126 |
| 9 | 57 | 40 | N/A | 0;127 |
| 0 | 48 | 41 | N/A | 0;129 |
| - | 45 | 95 | 31 | 0;130 |
| = | 61 | 43 | N/A | 0;131 |
| [ | 91 | 123 | 27 | 0;26 |
| ] | 93 | 125 | 29 | 0;27 |
| Spacebar | 92 | 124 | 28 | 0;43 |
| ; | 59 | 58 | N/A | 0;39 |
| ' | 39 | 34 | N/A | 0;40 |
| , | 44 | 60 | N/A | 0;51 |
| . | 46 | 62 | N/A | 0;52 |
| / | 47 | 63 | N/A | 0;53 |
| ` | 96 | 126 | N/A | 0;41 |
| Enter (keypad) | 13 | N/A | 10 | 0;166 |
| / (keypad) | 47 | 47 | 0;142 | 0;74 |
| * (keypad) | 42 | 0;144 | 0;78 | N/A |
| - (keypad) | 45 | 45 | 0;149 | 0;164 |
| + (keypad) | 43 | 43 | 0;150 | 0;55 |
| 5 (keypad) | 0;76 | 53 | 0;143 | N/A |
| F1 | 0;59 | 0;84 | 0;94 | 0;104 |
| F2 | 0;60 | 0;85 | 0;95 | 0;105 |
| F3 | 0;61 | 0;86 | 0;96 | 0;106 |
| F4 | 0;62 | 0;87 | 0;97 | 0;107 |
| F5 | 0;63 | 0;88 | 0;98 | 0;108 |
| F6 | 0;64 | 0;89 | 0;99 | 0;109 |

*(continued)*

### Table 6-5 ANSI.SYS Keyboard Control Codes *(continued)*

| *Key* | *Standard* | *With Shift* | *With Ctrl* | *With Alt* |
|---|---|---|---|---|
| F7 | 0;65 | 0;90 | 0;100 | 0;110 |
| F8 | 0;66 | 0;91 | 0;101 | 0;111 |
| F9 | 0;67 | 0;92 | 0;102 | 0;112 |
| F10 | 0;68 | 0;93 | 0;103 | 0;113 |
| F11 | 0;133 | 0;135 | 0;137 | 0;139 |
| F12 | 0;134 | 0;136 | 0;138 | 0;140 |
| Home | 0;71 | 55 | 0;119 | N/A |
| Up arrow | 0;72 | 56 | 0;141 | N/A |
| PgUp | 0;73 | 57 | 0;132 | N/A |
| Left arrow | 0;75 | 52 | 0;115 | N/A |
| Right arrow | 0;77 | 54 | 0;116 | N/A |
| End | 0;79 | 49 | 0;117 | N/A |
| Down arrow | 0;80 | 50 | 0;145 | N/A |
| PgDn | 0;81 | 51 | 0;118 | N/A |
| Insert | 0;82 | 48 | 0;146 | N/A |
| Delete | 0;83 | 46 | 0;147 | N/A |
| Print Screen | N/A | N/A | 0;114 | N/A |
| Pause | N/A | N/A | 0;0 | N/A |
| Backspace | 8 | 8 | 127 | 0 |
| Enter | 13 | N/A | 10 | 0 |
| Tab | 9 | 0;5 | 0;148 | 0;165 |
| Home (gray key) | 224;71 | 224;71 | 224;119 | 224;151 |
| Up arrow (gray key) | 224;72 | 224;72 | 224;141 | 224;152 |
| PgUp (gray key) | 224;73 | 224;73 | 224;132 | 224;153 |
| Left arrow (gray key) | 224;75 | 224;75 | 224;115 | 224;155 |
| Right arrow (gray key) | 224;77 | 224;77 | 224;116 | 224;157 |
| End (gray key) | 224;79 | 224;79 | 224;117 | 224;159 |
| Down arrow (gray key) | 224;80 | 224;80 | 224;145 | 224;154 |
| PgDn (gray key) | 224;81 | 224;81 | 224;118 | 224;161 |
| Ins (gray key) | 224;82 | 224;82 | 224;146 | 224;162 |
| Del (gray key) | 224;83 | 224;83 | 224;147 | 224;163 |

Notice that key combinations share the same codes as some single keys. Ctrl-H is the same as the Backspace key, for example. Try entering a command at the DOS prompt and press Ctrl-H; the previous character is removed, as if you pressed Backspace. Furthermore, some key combinations are not supported. DOS does not support Shift-Enter, for instance.

## How to remap keys

By using the ANSI.SYS p command, you can reassign the meaning of a key. The syntax of the p command is as follows:

`Esc[`*`keycode`*`;`*`newcode`*`p`

Replace the *keycode* variable with the scan code for the key you want to redefine. The *newcode* parameter represents the replacement key code. (Other uses of the p command are discussed in the next section.)

Here's an example: You can swap the semicolon and the colon keys so that the colon is the unshifted key and the semicolon is the shifted key. If you refer back to Table 6-5, you can see that the semicolon has the value 59, and the colon (which is normally Shift-semicolon) has the value 58. The following batch file, SWAPSEMI.BAT, swaps these keys:

```
@echo off
echo ←[59;58p
echo ←[58;59p
```

If the key is a standard ASCII keyboard character (that is, not an Alt or a Ctrl key combination, a function key, etc.), you can specify the key as a string instead of specifying the scan or ASCII code. Using string literals rather than scan codes makes a batch file much clearer. For example, you could write the batch file in the preceding example as follows:

```
@echo off
echo ←[';';':'p
echo ←[':';';'p
```

Usually, when you use ANSI.SYS to change keystrokes, only the characters typed at the command line are affected. Most programs, including EDIT, ignore such reassignments. Shucks.

## How to assign strings or commands to keys

You can use the p command to assign more than one character or keystroke to a key. You can specify multicharacter strings and ASCII code, provided that each entry is separated by a semicolon. The following ANSI.SYS command, for example, assigns the string *apple* to the uppercase A key:

```
echo ←['A';'Apple'p
```

When uppercase A is pressed, *Apple* is written to the keyboard buffer. It's best to avoid mapping the standard characters, such as letters of the alphabet and numbers because doing so can result in a lot of frustration. Imagine typing *A:* and seeing *Apple:* appear on the display.

The function keys, along with the Alt- and Ctrl- key combinations are good candidates for reassignment. The following batch file assigns the DOS HELP command to the F1 function key:

```
@echo off
echo ←[0;59;'help';13p
```

The 0;59 entry identifies the key to be remapped — F1. When F1 is pressed, ANSI.SYS substitutes the letters *help* followed by 13, which is the code for the Enter key. After the assignment is made, pressing F1 is the same as pressing the four characters *h,e,l,* and *p,* and then pressing Enter.

Similarly, the following batch file remaps the F2 key to launch EDIT:

```
@echo off
echo ←[0;60;'edit';13p
```

**CAUTION**

Do not assign more than 200 characters to a single key. ANSI.SYS only allocates 200 bytes for the key buffer. If you assign more characters, they have a nasty habit of overriding COMMAND.COM code. When this occurs, the system hangs, forcing a reboot.

**GEM**

A simple but effective antitampering device is to remap the Enter key when your system boots and then assign an unmap function to a secret key. The following extract from AUTOEXEC.BAT illustrates the approach:

```
echo ←[13;8p
echo ←[24;13p
echo ←[0;133;'prompt $e[13;13p$p$g';13;13p
echo ←[0;134;'prompt $e[13;8p$e[2J';13;13p
```

The first line assigns the Backspace function to the Enter key. When someone presses Enter, all that happens is that the last character on the command line is deleted. The second line assigns the Enter function to Ctrl-X; when you want to enter a command, you just press Ctrl-X.

The third line assigns to the F11 key a prompt command that reassigns the Enter command to the Enter key. After the system boots, you press F11 to set everything back to normal. The F12 key is assigned a prompt command that sets the Enter key as Backspace again. If you want to leave your office for a few minutes, you simply press F12. The Enter key is remapped, the screen is cleared, and the prompt disappears. When you return, press F11 and you are back in business. It's far from foolproof, but it deters the casual snoop.

## Going Beyond ANSI.SYS

The idea of providing a device driver to enhance the keyboard and display is a good one. But, with its roots in dumb mainframe terminals, ANSI.SYS doesn't go far enough. The folks at Hersey Micro Consulting Inc. developed an excellent replacement for ANSI.SYS, which is called FANSI-Console. Like ANSI.SYS, FANSI-Console provides a device driver (called FCONSOLE.DEV) that supports all the ANSI.SYS codes and a whole lot more.

FANSI-Console offers a variety of additional keyboard and display options, some of which follow:

- Sets the screen into super-dense mode, which supports up to 60 lines on VGA systems and even 50 lines on CGA
- Dramatically increases the screen-drawing speed
- Increases the size of the keyboard type-ahead buffer to 255 characters
- Supports a single-finger typing mode. Ideal for handicapped users, this mode enables the Ctrl and Alt keys to act as toggles affecting the next key typed. For example, you press the Alt key once and then press the X for Alt-X instead of holding down Alt and pressing X simultaneously.
- Sets Caps Lock to work as it does on a standard typewriter, where the Caps Lock key is disabled the next time the standard Shift key is pressed
- Provides a tactile clicking sound when keys are pressed
- Through a feature called *scroll recall,* enables you to look at text (such as output from DIR) that scrolled off the screen
- Enables keyboard reassignments to function within programs such as EDIT as well as on the command line

Because a Shareware version of FANSI-Console is included on Bob's Shareware Collection disk set, you can try it out for yourself.

## Summary

This chapter explained ways to use ANSI.SYS to enhance the display and keyboard. You learned the following points:

- You must load ANSI.SYS in CONFIG.SYS using the DEVICE or DEVICEHIGH statements.
- You send commands to ANSI.SYS using a special two-character signature that comprises the Esc and [ characters. You communicate commands via the ECHO, PROMPT, or TYPE commands.
- When writing batch files to communicate with ANSI.SYS, you should use the MEM and FIND commands to first ascertain whether ANSI.SYS is loaded in memory.
- Using the m command, you can control the foreground and background display colors and display attributes.
- ANSI.SYS provides commands for positioning the cursor as well as for saving and subsequently restoring the cursor position.
- Although you can use the p command to redefine the purpose of keys and assign commands to keys, be careful not to assign more than 200 characters to a key.

More and more devices are being incorporated into PCs, and more and more compatibility issues are surfacing. Chapter 7 explores the ways that hardware devices use IRQs, DMA channels, and the bus to integrate with the main system.

# Chapter 7

# Juggling IRQs, DMAs, Buses, Device Drivers, and Other "What's Its"

***In This Chapter***

- Looking under the hood
- Understanding interrupts and IRQ conflicts
- Understanding DMA
- Deciding which bus is best: ISA, MCA, or EISA
- Deciphering device drivers

## Looking under the Hood

Back in the good old days, computer users were blissfully ignorant of the PC's inner workings. They just installed the software and it worked. The most challenging task was to configure the dip switches on a new internal card to make sure that they were configured to use a unique LPT or COM port. Boy, have things changed.

With the constant desire to push PC hardware to its limits, conservative standards have been usurped, and everyone is getting "down to the metal" and accessing hardware directly. The latest software and hardware products do whatever it takes to get the best performance, such as bypass BIOS and access PC components directly. Although the upside is improved performance and greater capabilities, the downside is greater complexity and increased hardware conflicts.

Most PCs are now more than mere word processors and calculators. You can configure them with fax cards, modems, scanners, digitized tablets, tape-drive units, CD-ROMs, speakers, two monitors, sound cards, and network cards. With this array of hardware, it's no wonder that you have to look "under the hood" and understand how a PC uses IRQs and how DMAs and other mechanisms work to ensure that one product does not conflict with another.

## Understanding Interrupts and IRQ Conflicts

The brain of the personal computer is the *microprocessor,* or *central processing unit (CPU).* The microprocessor differs from most of the other chips on the system board because it can perform numerical computations.

The early Intel microprocessors were not adept at performing high-speed floating-point computations. Many users of math-intensive software (such as spreadsheet and CAD applications) installed a math (or numeric) coprocessor to improve performance. Software written to take advantage of the math coprocessor can instruct the microprocessor to pass floating-point calculations to the math coprocessor; it in turn returns the result. With the introduction of the 80486 microprocessor, Intel incorporated the math coprocessor into the main microprocessor chip.

Another unique feature of the microprocessor chip is that it is the system board's grand marshall: It controls the communications on the system board by passing data and signals to and from other devices.

Because many of the ancillary computing tasks are performed by supporting *controller* chips, the microprocessor can focus on the primary management and computational tasks. Some controllers are dedicated to serving the disk drives and the keyboard, for example, and some I/O controllers manage input and output to external devices. The organization is similar to an operating room, in which the microprocessor is the surgeon and many important (but subordinate) functions are performed by the support team, such as the anesthesiologist, the equipment technician, and nurses.

### Interrupting the microprocessor

The microprocessor must be notified when certain events occur or when a device needs attention. When you press the Ctrl-Break key combination, for example, the keyboard controller must inform the microprocessor immediately. Similarly, if you receive a fax while you are recalculating a spreadsheet, the microprocessor must ensure that the data coming in through a serial port can be saved to disk. These signals to the microprocessor for attention are called *interrupts.* Hundreds of interrupts can occur every second (even more interruptions than an author gets).

Interrupts are classified in two major categories: *hardware* and *software.* A hardware interrupt is generated by the microprocessor, another chip, or a device that needs the microprocessor's attention. Disk drive, keyboard, and system timer controllers all generate hardware interrupts. As its name suggests, a software interrupt is generated by a program. *Software interrupts* provide ways for a program to interact with DOS and ROM BIOS.

## Interrupt vectors

When a microprocessor is interrupted, it must suspend its current task, save all the data stored in the registers, and respond to the interrupt. When the interruption has been serviced, the saved data is restored to the registers, and the original task resumes.

The data saved before a hardware interrupt is serviced is stored on the stack. Every time a hardware interrupt occurs, DOS allocates a stack from a stack pool. With the potential for hundreds of interrupts per second, DOS can run out of stacks, or a stack can run out of space and generate a *stack overflow.* If you encounter these problems, use the STACKS command in CONFIG.SYS to increase the number and size of the interrupt stacks. See Part IV, "The Complete DOS 6 Command Reference," for more details about the STACKS command.

Every interrupt, whether it is a hardware interrupt or a software interrupt, is assigned a unique interrupt number. Instead of every interrupt shouting "Oy!" to get the CPU's attention, the interrupt shouts its assigned interrupt number ("Interrupt 10!" for example). When the microprocessor is interrupted, the code at a specific address (known as the *interrupt vector*) is executed. In Chapter 11, you learn how to use DEBUG, for example, to change the cursor shape by calling interrupt 10h (16 decimal). The code invoked in response to an interrupt is known as an *interrupt service routine,* or ISR.

Interrupt vectors can be redirected to point to different vectors. This technique is known as *installing an interrupt handler.* Normally, when a DOS critical error occurs, for example, an interrupt 24 is generated and DOS invokes the code that displays the infamous message to choose Abort, Retry, or Fail. By redirecting interrupt 24 to some custom code, you can avoid the ugly message and handle the error more gracefully. This approach is used in Chapter 11 to test whether a disk is in drive A without generating a critical-error message.

GEM

Every interrupt has a corresponding interrupt vector that is the *segment:*offset of the code that is executed when the interrupt occurs. The area of memory that stores the table of interrupt addresses is known as the *interrupt vector table.* At boot time, BIOS sets the location of the interrupt vector table to the beginning of RAM at address 0000:0000. The vector table contains one record for every interrupt, starting with interrupt 0 (CPU division by zero), followed by interrupt 1 (single-step execution), and so on. Every record in the table contains two words (a *word* is an unsigned integer that occupies two bytes of memory). The first word is the vector offset, and the second word is the vector segment. When an interrupt occurs, the corresponding record in the interrupt vector table is accessed, the offset value is loaded into the IP register, and the segment value is loaded into the CS register.

Not all entries in the interrupt vector table point to executable code. Some vectors point to data structures. Interrupt 41h, for example, points to the fixed disk drive parameter table.

## Interrupt request lines

Electronic anarchy would occur if the microprocessor had to handle every interrupt immediately. A special support chip, the *programmable interrupt controller (PIC),* is responsible for managing hardware interrupts. The PIC manages the queue of hardware interrupts and passes them one at a time to the CPU.

Every hardware device that needs to interrupt the microprocessor must be assigned an *interrupt request line,* often called an IRQ. The interrupt request line is a direct line to the PIC from the hardware device. In the original PC and XT systems, the PIC could accommodate a maximum of only 8 different IRQs; when the 80286-based systems were introduced, two PIC chips were used to expand this limit to 15.

If you think that 8, or even 15, IRQs is not a sufficient number, you're right. In addition to standard devices, such as the hard disk controller, the math coprocessor, the video system, IO ports, and the real-time clock, additional hardware devices, such as a fax card, mouse, sound board, and scanner, also want an IRQ.

Over time, de facto standards for assigning IRQs have evolved. Tables 7-1 and 7-2 show these standard IRQ assignments for 8086 and 80286+ systems, respectively.

**Table 7-1** **IRQ Assignments for 8086-/8088-Based Systems**

| *IRQ* | *Device Description* |
|---|---|
| 0 | IRQ timer |
| 1 | Keyboard |
| 2 | IRQ cascade |
| 3 | COM2 |
| 4 | COM1 |
| 5 | Hard disk controller |
| 6 | Floppy disk controller |
| 7 | LPT1 |

**Table 7-2** **IRQ Assignments for 80286-Based Systems and Higher**

| *IRQ* | *Device Description* |
|---|---|
| 0 | IRQ timer |
| 1 | Keyboard |
| 2 | IRQ cascade |
| 3 | COM2 and COM4, or bus mouse |
| 4 | COM1 and COM3 |
| 5 | LPT2 |
| 6 | Floppy disk controller |
| 7 | LPT1 |
| 8 | Real-time clock |
| 9 | Remapped IRQ from IRQ2 |
| 10 | Available |
| 11 | Available |
| 12 | Mouse (on IBM PS/2 systems) |
| 13 | Math coprocessor |
| 14 | Hard disk controller |
| 15 | Available |

## Connecting Two Interrupt Controllers

The programmable interrupt controller used in the original PCs and XTs was Intel's 8259A controller chip. Two 8259A chips are used in 80286+ systems, in which the first chip's IRQ2 line is redirected from the system bus to the second 8259A interrupt controller, and the real IRQ2 line is redirected to IRQ9. All interrupts received by the second controller are passed to the first controller by way of IRQ2 and then are passed to the microprocessor.

On PC and XT systems, the order of the IRQ numbers controls the interrupt priority. IRQ0 has a higher priority than IRQ1, and so on, through to IRQ7, with the lowest priority. The situation is a little more complicated on 80286+ systems. Because all interrupts on the second chip are redirected to IRQ2 on the first chip, all the interrupts on the second chip (IRQ8 through IRQ15) have priority over the interrupts in the range from IRQ3 to IRQ7.

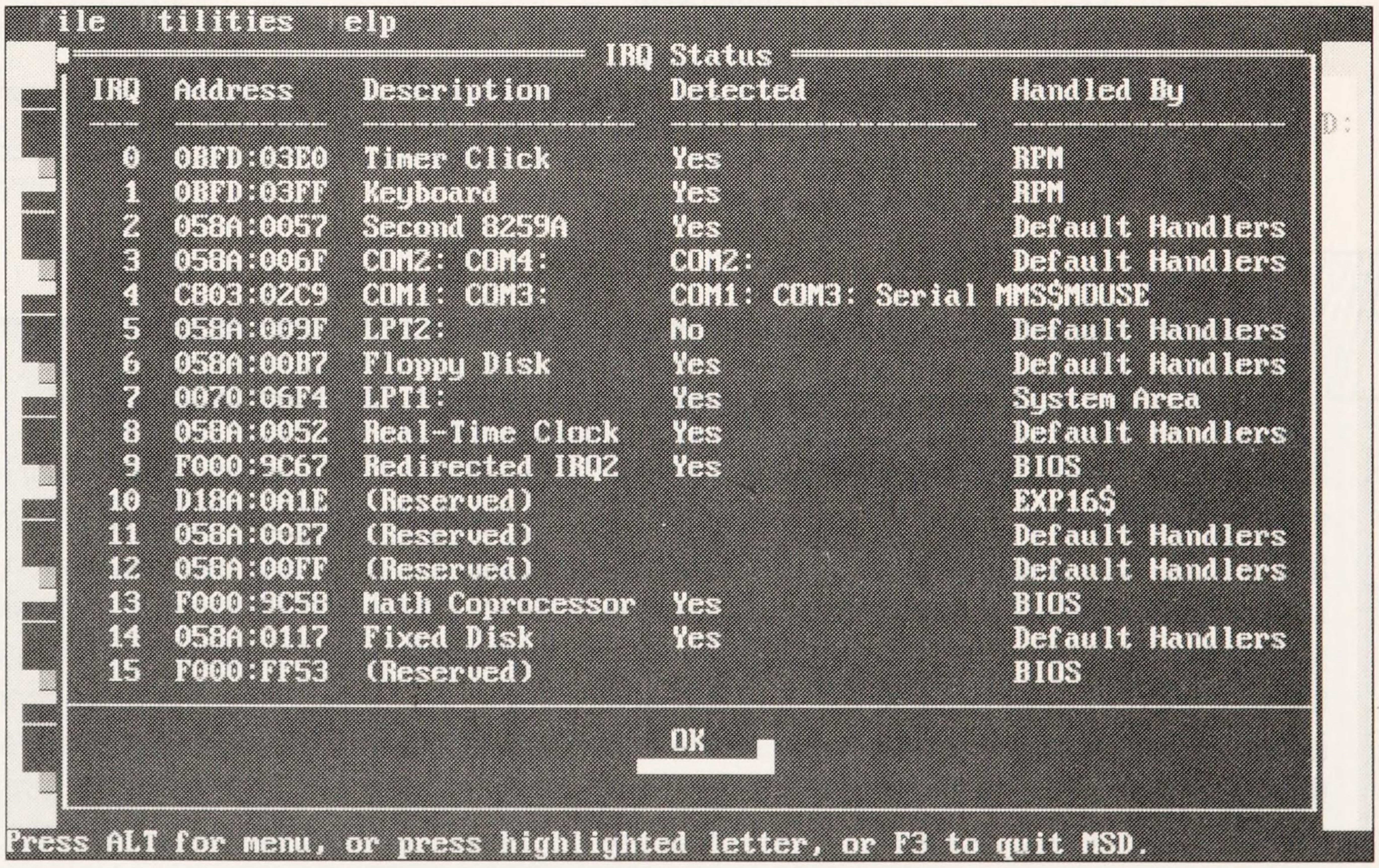
## Reviewing your IRQ assignments

DOS 6 includes the invaluable utility MSD (for Microsoft System Diagnostics), which can help you to ascertain your current IRQ use. Figure 7-1 shows a typical display that is generated when you execute MSD and click on the IRQ button, or press Q, to select it.

The second column details the interrupt vector (the code that is executed when the interrupt occurs). The third column contains a generic description of the IRQ port; the third and fourth columns show whether the IRQ line is in use and, if so, the corresponding program or device.

Any free lines have an entry of No in the Detected column, or the column is empty. Not all IRQs with an empty Detected column, however, are available. In Figure 7-1, for example, IRQ10 is not detected, and yet it is used by the network card.

To make sure that an IRQ line is available, check the third *and* fourth columns. If the Handled By column does not say BIOS or Default Handlers, it is probably in use. In the example, the network device driver EXP16$ is identified as the handler for IRQ10.



```
                              IRQ Status
IRQ  Address    Description          Detected            Handled By
---  ---------  -------------------  ------------------  ----------------
  0  0BFD:03E0  Timer Click          Yes                 RPM
  1  0BFD:03FF  Keyboard             Yes                 RPM
  2  058A:0057  Second 8259A         Yes                 Default Handlers
  3  058A:006F  COM2: COM4:          COM2:               Default Handlers
  4  C803:02C9  COM1: COM3:          COM1: COM3: Serial MMS$MOUSE
  5  058A:009F  LPT2:                No                  Default Handlers
  6  058A:00B7  Floppy Disk          Yes                 Default Handlers
  7  0070:06F4  LPT1:                Yes                 System Area
  8  058A:0052  Real-Time Clock      Yes                 Default Handlers
  9  F000:9C67  Redirected IRQ2      Yes                 BIOS
 10  D18A:0A1E  (Reserved)                               EXP16$
 11  058A:00E7  (Reserved)                               Default Handlers
 12  058A:00FF  (Reserved)                               Default Handlers
 13  F000:9C58  Math Coprocessor     Yes                 BIOS
 14  058A:0117  Fixed Disk           Yes                 Default Handlers
 15  F000:FF53  (Reserved)                               BIOS

Press ALT for menu, or press highlighted letter, or F3 to quit MSD.
```

**Figure 7-1**: MSD displays the system's IRQ configuration.

As Table 7-2 illustrates, some devices share an IRQ line. Both COM1 and COM3, for example, are assigned to IRQ4. IRQ lines are a popular item. The PC engineers believed that, because no one is likely to use both COM1 and COM3 simultaneously, it makes sense for COM1 and COM3 to share a line. After all, who would run two modems simultaneously from one PC? A potential for conflict does exist, however, when one port is used for a mouse and its sister port is used for a modem. A mouse and a modem are likely to be in use simultaneously. If you have a mouse and a modem attached, make sure that they use COM ports on different IRQ lines.

Every hardware device normally needs a dedicated IRQ line. If you install a fax card, for example, you must configure the card to use a specific IRQ.

Before you buy new hardware, make sure that your system can accommodate it. You must determine, specifically, whether your system has an available IRQ line and whether the new hardware can use the IRQ line. Eight-bit boards, and even some 16-bit boards, can access only the first eight IRQs. Check with your hardware vendor to determine for which IRQs their hardware can be configured.

To accommodate a new hardware device, you may have to disable an unused COM or LPT port, or even reconfigure existing devices to use different IRQs, so that a new, less flexible device can be assigned an appropriate IRQ line.

## Understanding DMA

The microprocessor, arguably the busiest chip in the PC, is clearly a performance bottleneck. The more tasks other chips can handle, the better.

One common and time-consuming task is the transfer of data to and from memory. Not all memory operations have to be managed by the microprocessor. When data is being transferred directly from the hard disk to a floppy disk, for example, no computations have to be performed, and the microprocessor therefore does not have to be directly involved. A special chip called the *direct memory access (DMA)* controller can transfer data directly to and from memory. Although the microprocessor does not perform the memory transfer, it still manages or oversees the memory transfer, and its burden is reduced significantly.

The original PC- and XT-compatible systems used a single Intel 8237A-5 DMA controller chip that provided four DMA channels. In theory, every channel can be used concurrently to move data to and from memory. One of these channels is dedicated to refreshing system memory, which leaves three channels for general-purpose memory moves. This chip unfortunately can handle only 8-bit memory transfers and can address only one megabyte of memory.

A second 8237A chip was added to 80286 and subsequent systems. Like the twin-interrupt controller configuration discussed earlier in this chapter, the second DMA controller hooks into one of the first DMA controller's channels and provides, therefore, a total of seven channels. The second chip can handle 16-bit memory transfer and therefore provide access to the full memory potential of the 80286 system and improve overall performance.

Generally, only one DMA channel at a time is used. The most notable exception is in backup software, in which two channels often are used simultaneously. One channel manages memory transfers to the hard disk, and the other channel manages memory transfers to the floppy disk. The concurrent use of two DMA channels often is referred to as *concurrent read-write* or *dual DMA*.

Some third-party backup systems offer three modes of backup speed:

- **High speed.** The DMA controller chip performs concurrent read-write operations.
- **Medium speed.** The DMA chip is used for memory transfers, but only one channel at a time is used.
- **Low speed.** The DMA controller is not used, and the backup program uses the microprocessor for standard I/O transfers.

The new DOS 6 backup programs MSBACKUP and MWBACKUP support, unfortunately, only medium-speed (single DMA) and low-speed options.

The memory manager EMM386 includes the D switch to control the size of the DMA buffer. The DMA buffer size, set to 16K by default, normally is appropriate. If you encounter the message that the DMA buffer size is too small or something similar, use the D switch to increase the buffer size. Any (kilobyte) value in the range from 16 to 256 is allowed.

## Which Bus Is Best: ISA, MCA, or EISA?

One of the primary reasons that the IBM PC became so popular (other than the IBM brand name) was that the PC was designed with an open architecture. The most tangible aspect of the open architecture was the use of generic expansion slots connected to the microprocessor and ancillary circuitry by a bus. Independent vendors could create expansion cards, such as internal modems, video display cards, and memory cards, that could plug in to the system board and therefore expand the system's capabilities. This open architecture ironically led to the fierce competition that forced IBM to implement a new bus design known as the *Micro Channel Architecture.* IBM chose to move away from its open architecture policies of the past and tried to impose large royalties on would-be copycats.

## The role of the bus

The bus is the computer's main highway that links all expansion cards to the microprocessor. The bus is a series of parallel "wires," called *circuit pathways.* As on a a multilane freeway, data moves down every pathway simultaneously. Generally, the more pathways, the greater the throughput and the better the performance. When you plug in an expansion card on the system board, the circuitry on the expansion card is connected directly to the bus.

The primary purpose of the bus is to transfer data to and from the microprocessor or from device to device by way of the DMA controller. With all expansion cards sharing the same pathways, data must be moved in a regimented and controlled manner. The bus is divided into these four main parts:

- Power lines
- Control lines
- Address lines
- Data lines

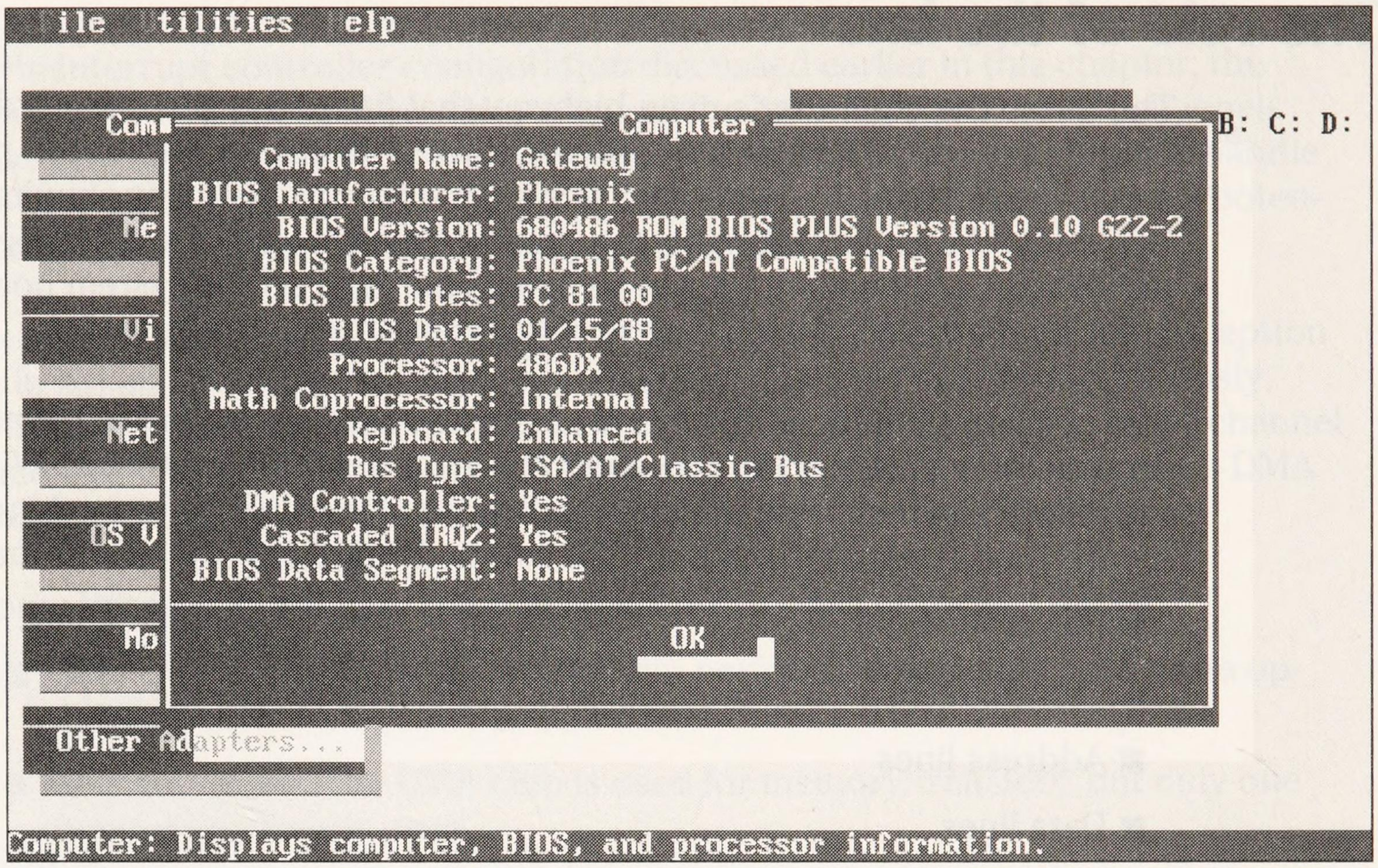
Not surprisingly, the power lines provide power to the expansion cards. The control lines are used to transmit timing signals from the system clock and to signal interrupts. Before any data is transmitted, the destination address is transmitted along the data lines and therefore alerts the receiving location that some data is ready to be shipped to it. The microprocessor then checks to see whether there is a ready signal (on the I/O Channel Ready line). If all is well, the data is transmitted along the data lines.

The number of lines dedicated to the *address* bus dictate the amount of total addressable memory. Twenty address lines, for example, can access one megabyte of memory. The number of bus lines dedicated to the *data* bus dictates the overall data throughput of the bus. As a rule, optimum performance is achieved when the number of data lines corresponds with the data lines in the microprocessor. The number of data lines is the most quoted characteristic of a PC bus. A 16-bit bus, for example, means that the bus uses 16 data lines.

In the PC world, buses come in three main flavors: ISA, MCA, and EISA. You can determine your system's bus flavor by running MSD and pressing C to display the computer settings (see Figure 7-2).

## The ISA bus

Naming PC bus architectures is a little like naming cough drops or colas. When a company starts selling a new cherry-flavored cough drop, the company feels compelled to give the old cough drop some sort of name, such as *original flavor*



```
     Computer Name: Gateway
 BIOS Manufacturer: Phoenix
      BIOS Version: 680486 ROM BIOS PLUS Version 0.10 G22-2
     BIOS Category: Phoenix PC/AT Compatible BIOS
     BIOS ID Bytes: FC 81 00
         BIOS Date: 01/15/88
         Processor: 486DX
  Math Coprocessor: Internal
          Keyboard: Enhanced
          Bus Type: ISA/AT/Classic Bus
    DMA Controller: Yes
     Cascaded IRQ2: Yes
 BIOS Data Segment: None
```

Computer: Displays computer, BIOS, and processor information.

**Figure 7-2:** The MSD Computer option shows the system bus type.

or *classic.* Originally, the bus used in PCs and XTs didn't have a name. There was only one bus type, so it didn't need a special distinguishing name. After the introduction of alternative PC bus architectures, the old-fashioned PC bus architecture was awarded the name *Industry-Standard Architecture,* or ISA. To be accurate, the bus used in the IBM PC AT (80286) defines the ISA bus.

The bus used in the first PCs had 62 lines; 20 of them were address lines, and 8 were data lines. The bus was limited to accessing 1MB of memory, and data was transferred in single-byte (8 bit) packets. The bus clock speed was the same as the microprocessor clock speed (4.77 MHz).

The original bus design clearly was conservative. While the PC engineers were developing the IBM-PC AT to use an 80286 processor, they expanded and enhanced the bus. The most obvious change in bus design was the installation of additional address and data lines.

To preserve compatibility with the original PC bus and therefore ensure that all (or most) existing expansion cards would work in the new bus, the additional lines were accessed through a second connector on the system board. The original bus used a 3.2-inch connector with 31 contacts along each edge. The AT bus supported a second 2-inch connector that provided an additional 16 contacts along each edge (see Figure 7-3).

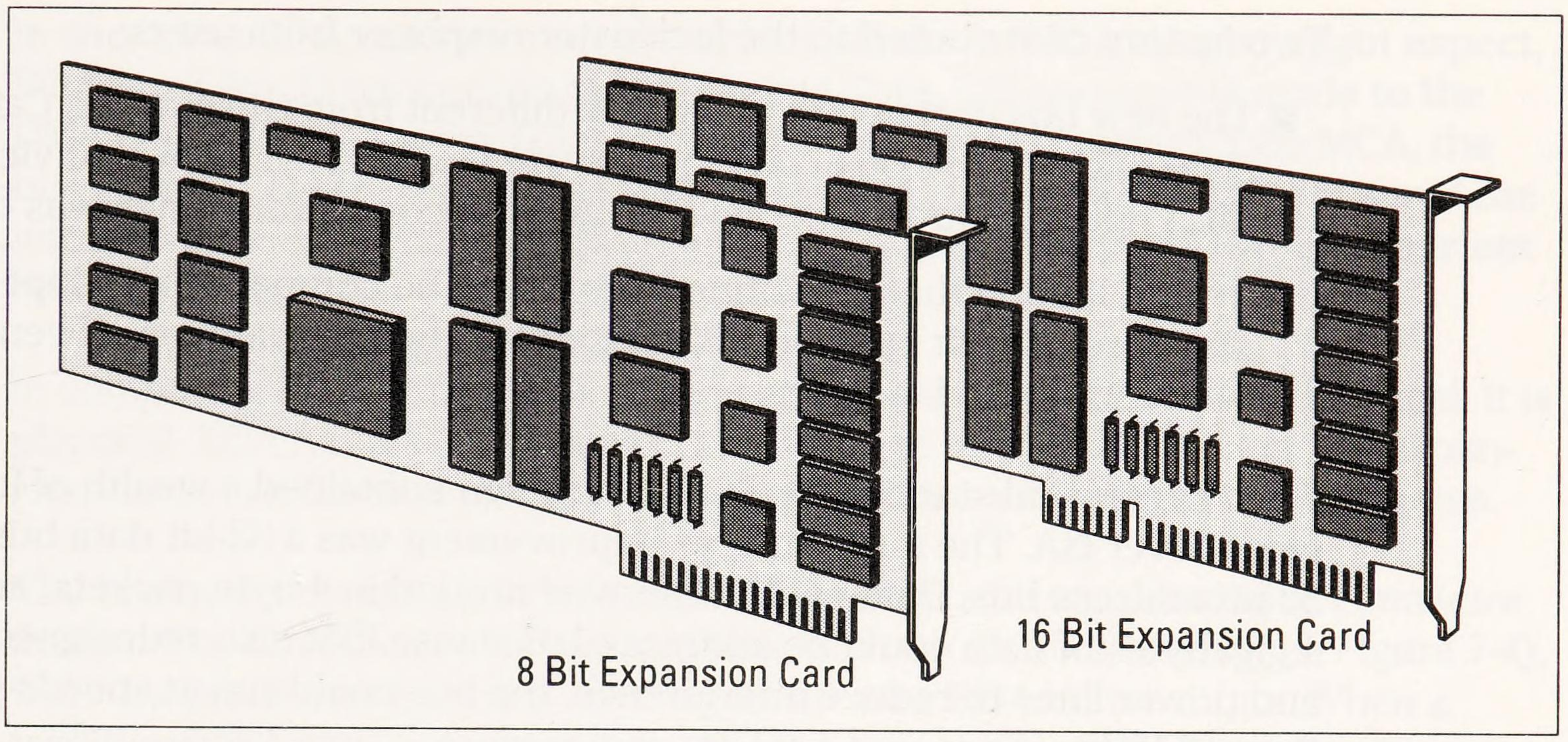


**Figure 7-3:** The AT bus used a second connector block to maintain backward compatibility with 8-bit expansion cards designed for the original PC.

This second connector provided four additional address lines, which made the total 24 address lines. A total of 16 megabytes of memory (the same as the 80286 processor) therefore could be addressed. Because eight data lines were added, data could be moved 16 bits at a time.

The engineers also made an important change to the bus timer design. The bus and the microprocessor originally used the same timer chip and therefore were forced to run at the same speed. Although the bus could run comfortably at 6 or 8 MHz, it could not keep pace at 12 or 16 MHz — the projected full speed of the 80286. In anticipation of ever-increasing microprocessor speeds, the AT bus was designed to run from a separate timer chip, which enabled it to run at a different (slower) speed than did the microprocessor.

## The MCA bus

In the mid-1980s, the ISA bus seemed to be the system bottleneck. Memory chips and microprocessors were getting increasingly faster, but the bus wasn't. IBM decided to release a family of computers, the PS/2, that used a completely new bus architecture designed to radically improve system throughput and engineered to accommodate microprocessor advances for at least ten years. IBM called this new bus the *Micro Channel Architecture* (MCA).

The first PCs to sport this new bus were released in 1987. IBM anticipated that, thanks to performance improvements and the IBM name, customers would demand MCA-based systems. MCA has yet to dominate the marketplace, however.

Two factors contributed to the lackluster response from users:

- The new bus design was physically different from the ISA bus. Cards designed for the ISA bus could not be used in MCA systems, and vice versa. MCA adapter cards were smaller, and the connector design was different.
- In anticipation that its competitors would be compelled to adopt the MCA standard in their systems, IBM imposed a hefty royalty on all vendors who built MCA-based systems.

From a technical standpoint, the MCA design contained a wealth of improvements over ISA. The most notable improvement was a 32-bit data bus and a 32-bit address bus. Data could be moved around in 4-byte packets, and a huge 4 gigabytes of data could be addressed. Because IBM also redesigned the signal and power lines to reduce interference, the bus could run at speeds greater than 10 MHz.

Significantly, the MCA bus included features that enable independent devices to take cooperative control of the bus. The concept of enabling another device to take control of the bus from the microprocessor is known as *bus mastering,* which enables two or more independent devices to communicate and not involve the microprocessor. Additionally, the bus-mastering design enabled MCA systems to support multiple (parallel) microprocessors (a system may have four 386 microprocessors installed on an expansion card, for example). The utilization of parallel microprocessors unfortunately presents major software-development challenges; no mainstream applications take advantage of parallel processors.

Another innovation in the MCA bus was the abolition of expansion-card dip switches. The technique that was used is similar to the way a PC can remember the hard disk and memory configuration by using battery-sustained CMOS memory. An MCA system can remember its expansion-card configuration. This neat feature enables you to configure expansion cards by using software and the keyboard rather than a screwdriver.

Clearly, the MCA architecture is technically superior to ISA.

## The EISA bus

While IBM was pushing the all-new MCA bus, the other major PC vendors (including such giants as Compaq, Hewlett Packard, and AST Research) decided to provide an alternative high-performance bus. After seeing the adverse consumer reaction to the lack of compatibility between ISA and MCA expansion cards, a primary goal was to ensure that the new bus would be fully compatible with ISA expansion cards. The result was the *Extended Industry Standard Architecture (EISA).*

In short, the EISA bus has a set of features similar to MCA. As you might expect, EISA provides 32 address lines and 32 data lines. Enhancements made to the DMA controllers allowed 32-bit DMA-driven memory transfers. Like MCA, the bus speed is controlled by a separate bus clock, which enables the bus and the microprocessor to run at different speeds. EISA also supports the all-important bus mastering.

In comparing MCA and EISA, the most significant difference is not electrical; it is physical. EISA has a major marketing advantage over MCA because ISA expansion cards can be used in EISA-based systems but not in MCA-based systems.

The backward compatibility is achieved through a clever yet simple connector design. The EISA connector is organized into two distinct levels (see Figure 7-4), with the bottom-level “teeth” serving as the new EISA connections. When a traditional ISA card is pushed into the EISA bus, the teeth do not extend low enough to make contact with the EISA-specific connections.

CAUTION

Although an EISA board physically fits in an ISA system, you should never insert one. EISA boards are not compatible with ISA buses, and you can damage both the system board and the EISA board.

Figures 7-5, 7-6, and 7-7 are examples of typical ISA, EISA, and MCA expansion boards, respectively. The ISA and EISA cards are similar, with the EISA

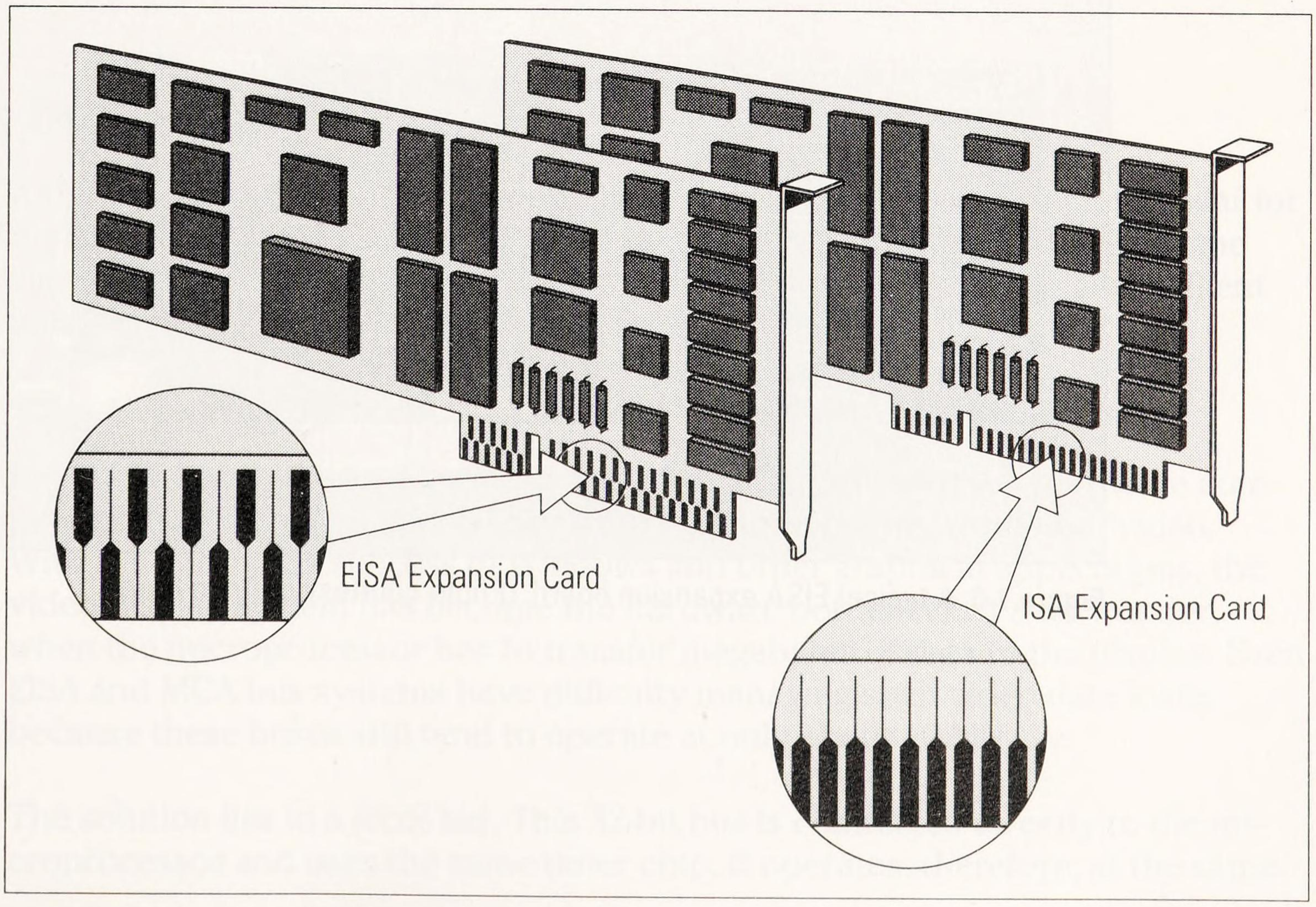


**Figure 7-4:** EISA expansion boards use two-tier connections compared with the single connection on ISA boards.

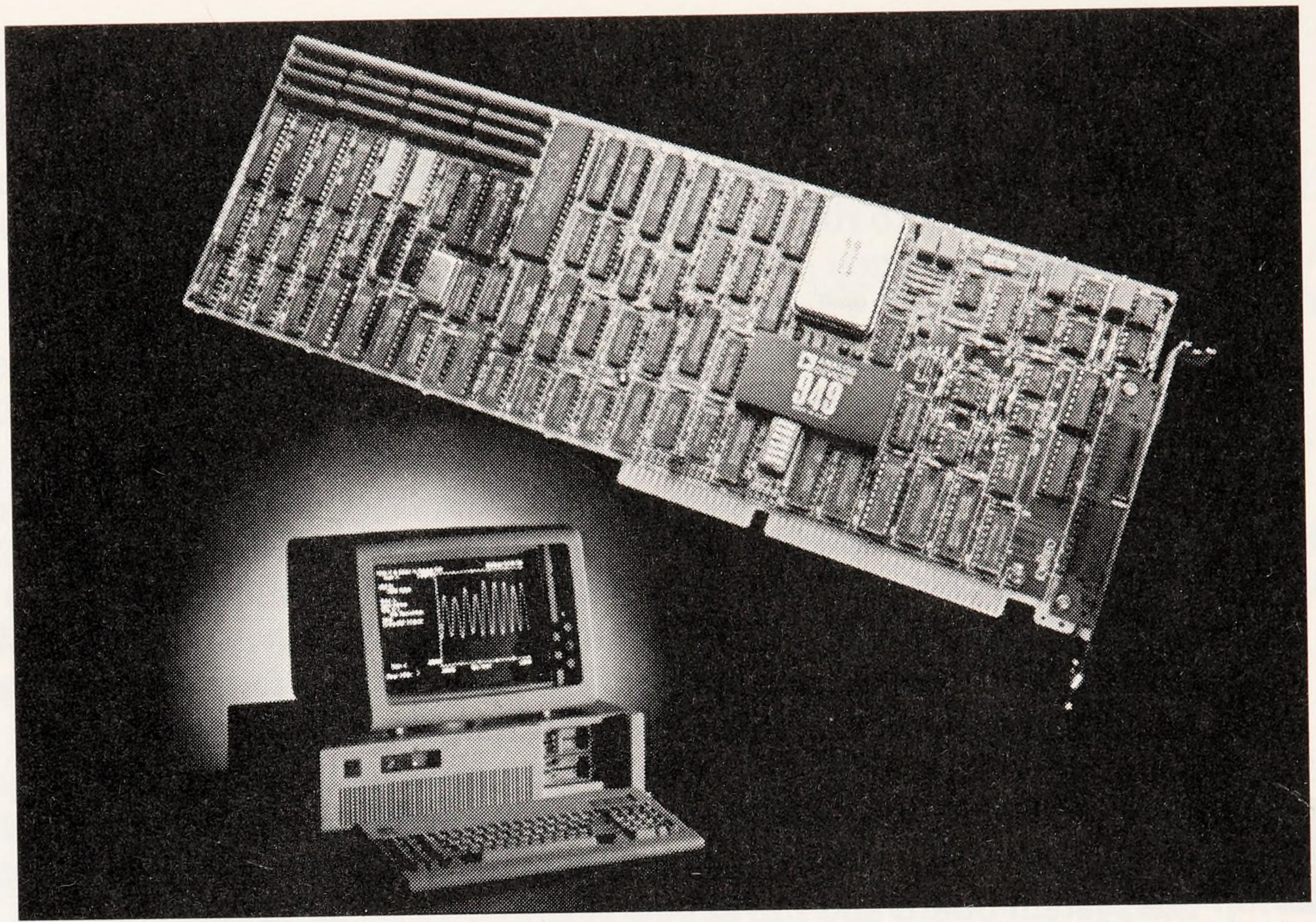


Figure 7-5: A typical 16-bit ISA expansion board. (Photo courtesy of Analog Devices Inc.)

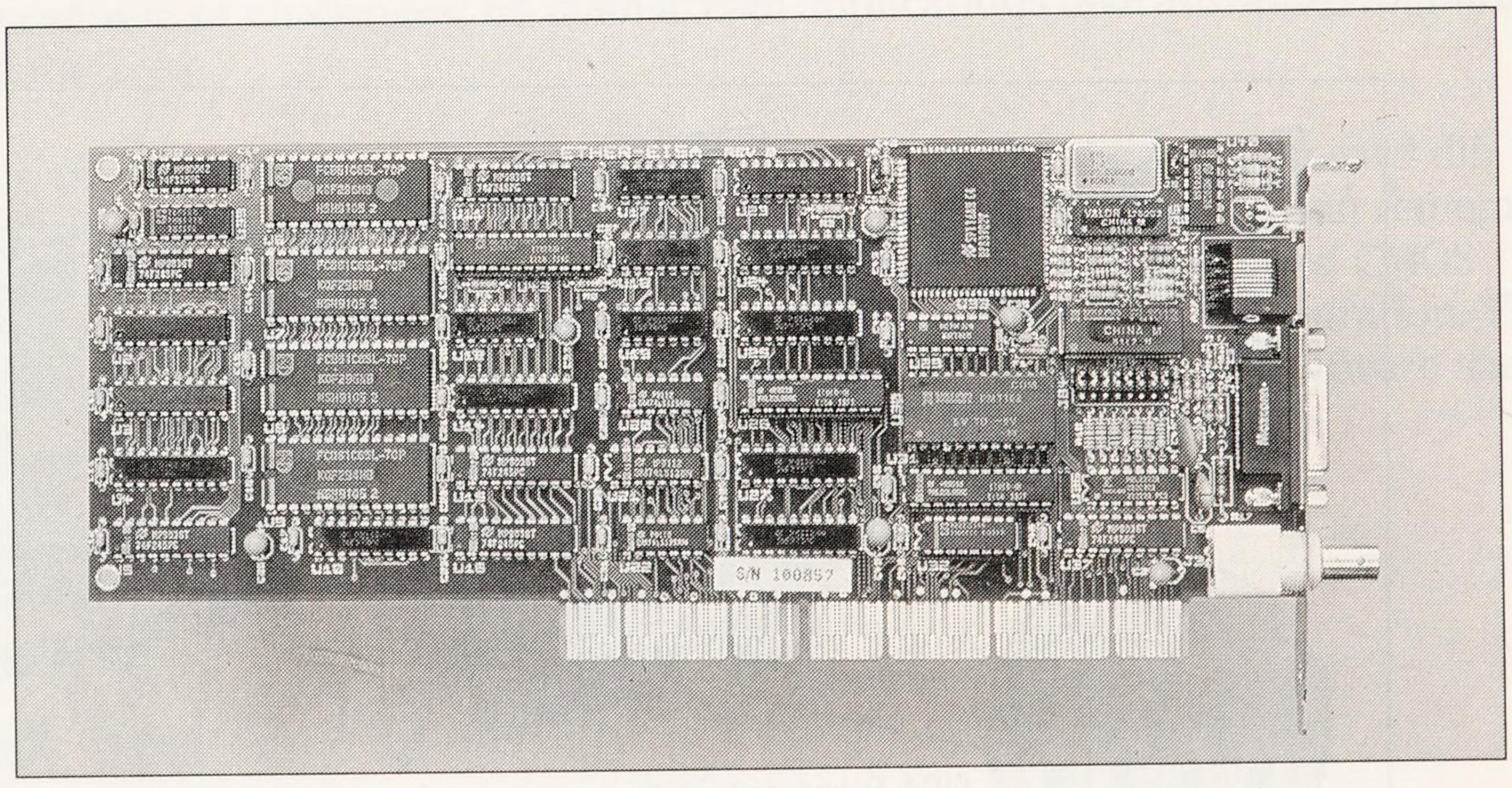


Figure 7-6: A typical EISA expansion board. (Photo courtesy of Anco Corp.)

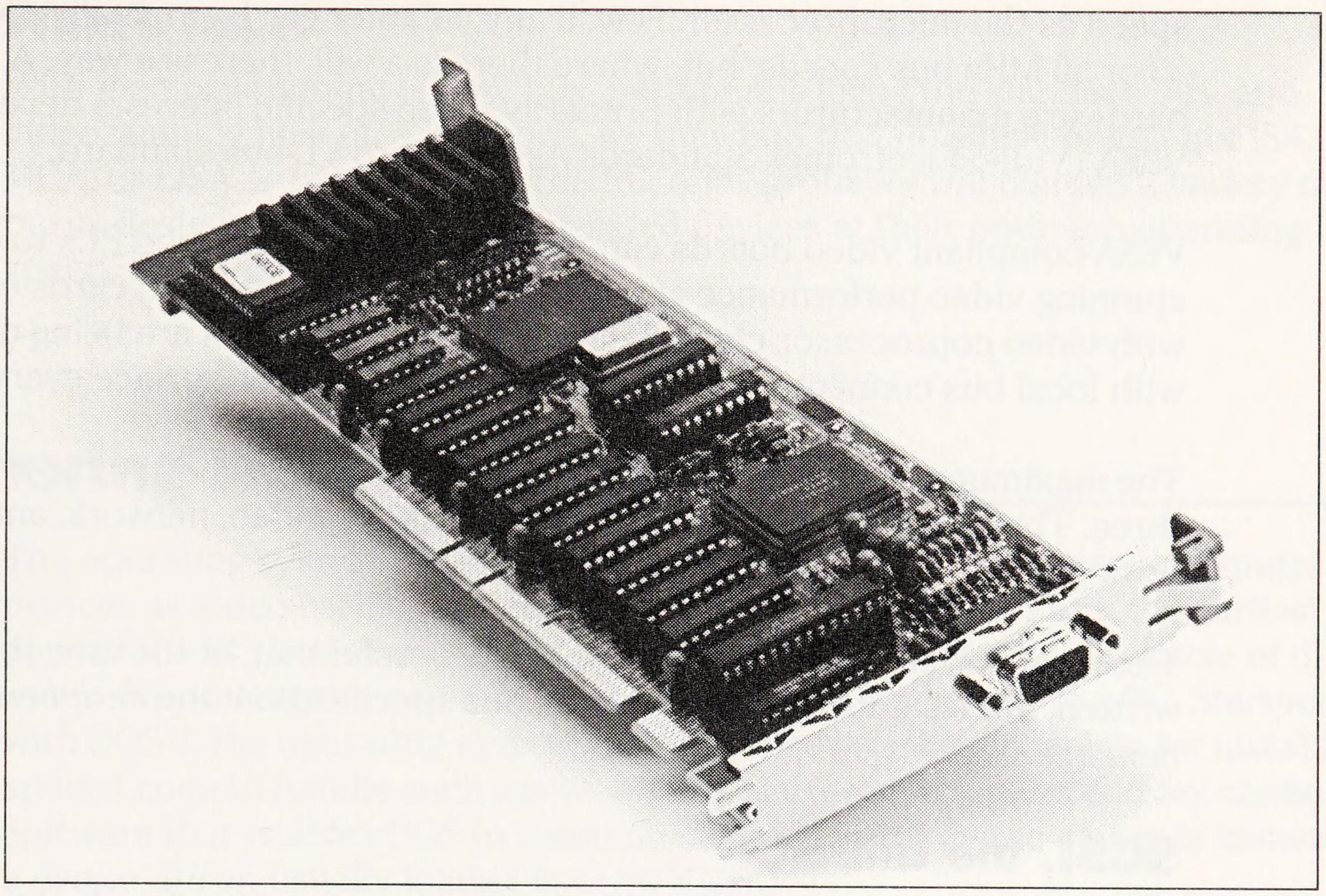


**Figure 7-7:** A typical MCA expansion board. (Photo courtesy of Genoa Systems Corp.)

connectors being slightly larger, but the MCA card is completely different in size, shape, and its unique connector design.

## Recent bus innovations

Although the MCA and EISA vendors have engaged in a major marketing war for dominance, the ISA bus has been quietly enhanced and improved with some significant and powerful innovations. ISA is proving to be much more resilient and hardy than many PC engineers anticipated.

### The local bus

If you had conducted a survey recently that asked users which hardware component most needed a speed boost, the resounding reply would be "video." With the enormous success of Windows and other graphical applications, the video display system has become the hardware bottleneck. It's no wonder, when the microprocessor has to transfer megabytes of data to the display. Even EISA and MCA bus systems have difficulty managing such video data loads because these buses still tend to operate at only about 10 MHz.

The solution lies in a *local bus*. This 32-bit bus is connected directly to the microprocessor and uses the same timer chip. It operates, therefore, at the same speed as the microprocessor. Few, if any, adapter cards originally could handle 33 or 50 MHz bus speeds; but, where there is a will, there is a way. A band of hardware manufacturers with primarily video-specific interests developed the VESA (Video Electronics Standards Association) VL-bus standard.

VESA-compliant video boards can plug into VL-bus-equipped PCs to obtain stunning video performance at a fraction of the cost of video cards equipped with video coprocessor chips. Increasingly more ISA PCs are being developed with local bus connectors, and new VESA video boards surface every month.

The maximum number of devices that now can be attached to the local bus is three. The primary candidates for these slots are video, network, and storage devices.

The local bus engineering has been so successful that, at the time this book was written, Intel is finishing its own local bus specification, the *Peripheral Component Interconnect (PCI).*

## SCSI, the UnBus

"When is a bus not a bus?" Yoda asked. "When it's a SCSI," replied the DOS Jedi.

The *Small Computer System Interface (SCSI),* pronounced "skuzzy," is a high-speed system for connecting such devices as hard disks, CD-ROMs, and tape drives. Many people think of the SCSI controller as just another hard disk controller, but it is much more.

Several devices (to a maximum of seven) all are connected in parallel by using a single-ribbon cable that in turn is connected to a SCSI controller. The SCSI cable is really an expansion bus for connecting high-speed storage devices and scanners. SCSI devices can communicate and transfer data between themselves and not involve the microprocessor or use the standard system bus. Imagine being able to save hard disk files to tape and not slow down the system. The SCSI design can work with any of the three ISA, MCA, and EISA standard bus types.

The original SCSI design used a 50-bit (or -line) bus and had 8 data bits. The SCSI-2 specification released in 1991 additionally supported 16 and 32 data lines that sported impressive data-transfer rates of more than 40MB per second. Even with the revised standard, SCSI is not free from problems. Not all SCSI devices cooperate with each other, and two detailed programming specifications currently vie for acceptance: the *Common Access Method Committee (CAM)* and the *Advanced SCSI Programming Interface (ASPI).*

Engineers are developing SCSI controllers that will attach to an ISA local bus. That's performance. Who said ISA was dead?

### What's next?

With memory running on one bus, peripherals using the standard bus, and video running now on a local bus, we may find that no single bus — not ISA, MCA, or EISA — will dominate the PC. A PC probably will contain a variety of buses designed to interconnect related devices at their optimum operating speed. Only time will tell.

## Deciphering Device Drivers

The operating system is designed to accommodate such standard computer devices as video cards, printers, modems, and disk drives. From the earliest days of the PC, however, it was obvious that DOS would not be capable of directly supporting the myriad devices that can be connected to a PC. Starting with DOS 2, the operating system provided a software mechanism for installing special code to handle such nonstandard devices as scanners and fax cards. Software that enables DOS to communicate with a hardware device is known as a *device driver,* usually loaded in CONFIG.SYS.

Many device drivers mimic standard devices. A hard disk device driver for a nonsupported hard disk, for example, responds to DOS's disk instructions in a standard way but uses different commands to communicate in turn with the hard disk. Similarly, a joystick device driver might make the joystick appear like a keyboard to DOS. The device driver acts as a translator, or go-between, to enable a foreign device to speak DOS's language, and vice versa.

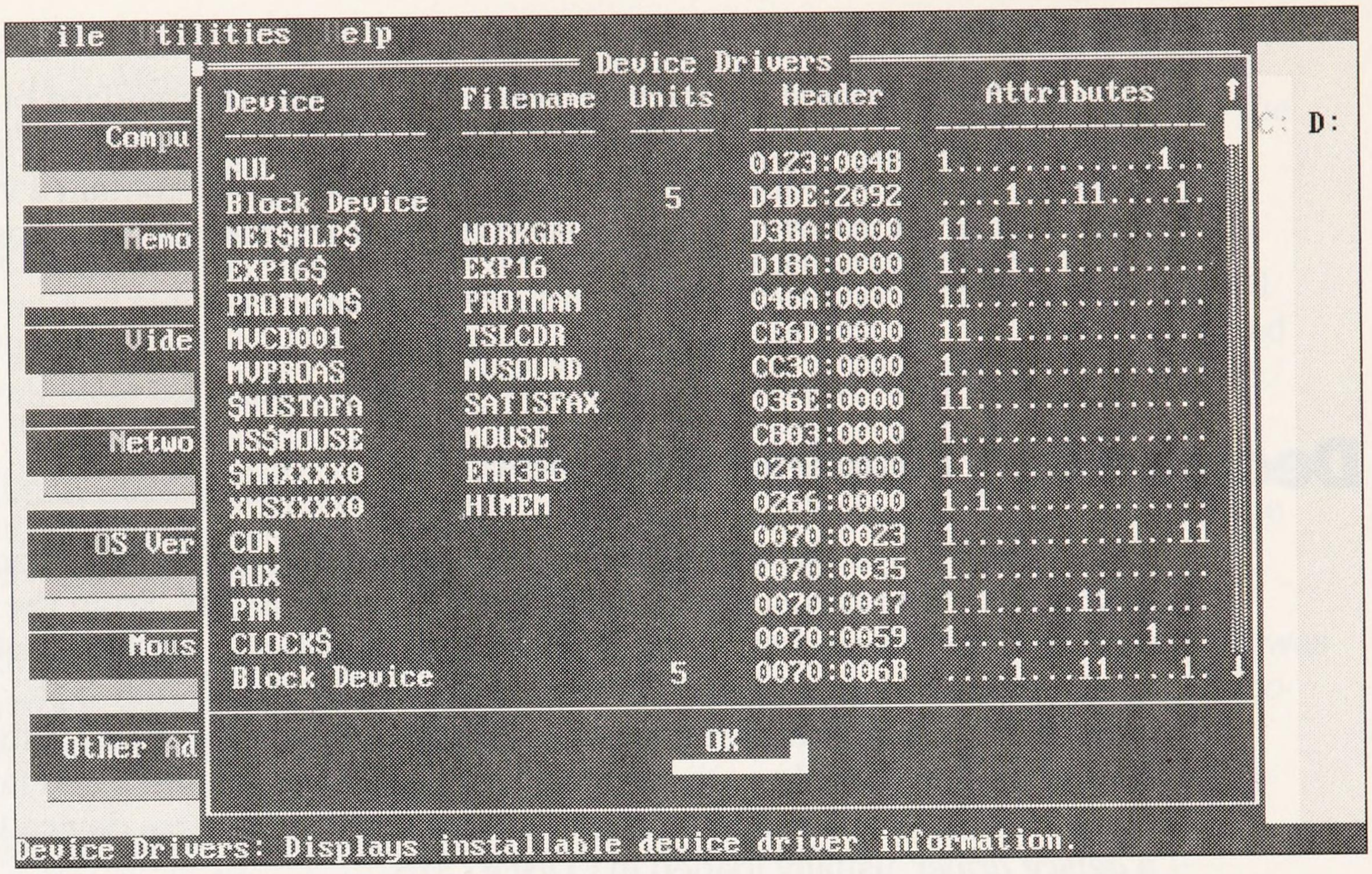
Behind the scenes, many standard DOS devices are supported by device drivers that are loaded from the hidden DOS file IO.SYS.

You execute MSD and press R to see a list of the device drivers installed on your system. Figure 7-8 shows a typical device driver list.

Device driver routines normally are invoked by a software interrupt. When a device driver is installed, it adds in the interrupt vector table an entry that points to the device driver's main code. Column 4 of the MSD device driver listing displays the starting address of the main device driver code. DOS maintains a chain, or list, of device drivers, in which each device driver points to the next one in the list. The first device driver in the chain is the NUL device.

You can display additional device-driver details, such as the driver size, by using the MEM /D command. To filter out superfluous data, use this command in conjunction with the find filter:

```
mem /d | find "device" /i
```



```
                        Device Drivers
Device          Filename  Units   Header      Attributes
------------    --------  -----   ---------   ----------------
NUL                               0123:0048   1............1..
Block Device                5     D4DE:2092   ....1...11....1.
NET$HLP$        WORKGRP           D3BA:0000   11.1............
EXP16$          EXP16             D18A:0000   1...1..1........
PROTMAN$        PROTMAN           046A:0000   11..............
MVCD001         TSLCDR            CE6D:0000   11..1...........
MVPROAS         MVSOUND           CC30:0000   1...............
$MUSTAFA        SATISFAX          036E:0000   11..............
MS$MOUSE        MOUSE             C803:0000   1...............
$MMXXXX0        EMM386            02AB:0000   11..............
XMSXXXX0        HIMEM             0266:0000   1.1.............
CON                               0070:0023   1..........1..11
AUX                               0070:0035   1...............
PRN                               0070:0047   1.1.....11......
CLOCK$                            0070:0059   1...........1...
Block Device                5     0070:006B   ....1...11....1.
```

Device Drivers: Displays installable device driver information.

**Figure 7-8:** MSD displays installed device drivers.

The command produces a listing similar to the following:

```
                        CON        System Device Driver
                        AUX        System Device Driver
                        PRN        System Device Driver
                        CLOCK$     System Device Driver
                        A: - E:    System Device Driver
                        COM1       System Device Driver
                        LPT1       System Device Driver
                        LPT2       System Device Driver
                        LPT3       System Device Driver
                        COM2       System Device Driver
                        COM3       System Device Driver
                        COM4       System Device Driver
     1088    (1K)       XMSXXXX0   Installed Device=HIMEM
     3104    (3K)       $MMXXXX0   Installed Device=EMM386
     4016    (4K)       $MUSTAFA   Installed Device=SATISFAX
      112    (0K)       PROTMAN$   Installed Device=PROTMAN
    17072   (17K)       MS$MOUSE   Installed Device=MOUSE
     9136    (9K)       MVPROAS    Installed Device=MVSOUND
    12720   (12K)       MVCD001    Installed Device=TSLCDR
     8928    (9K)       EXP16$     Installed Device=EXP16
     4352    (4K)       NET$HLP$   Installed Device=WORKGRP
```

Device drivers are in two basic categories: character drivers and block drivers. Character device drivers are single-byte-oriented devices, such as printer ports. These devices include the standard DOS-supplied devices, such as AUX, CON, LPT, and NUL. Block devices manage blocks of data and usually are used for disks, CD-ROMs, and other data-intensive drivers. Every block device is assigned a logical drive letter. The rightmost column, labeled Attributes, in MSD’s device driver window provides more information about the device driver format. Table 7-3 lists the meaning of every attribute.

Device drivers do not have to be oriented to a specific hardware device. ANSI.SYS, for example, is a device driver that enhances the capabilities of the standard keyboard and display. Also, you can load some device drivers as TSRs rather than as standard devices in CONFIG.SYS. In the most notable example, the Microsoft Mouse driver, you can either load MOUSE.SYS in CONFIG.SYS or execute MOUSE.COM from the command line.

To save conventional memory, you can use DEVICEHIGH and LOADHIGH to load device drivers and TSRs, respectively, into upper memory. Refer to Chapter 5 for more information about loading device drivers HIGH.

**Table 7-3 Device Driver Attributes**

| *Bit Position* | *Attribute* | *Meaning* |
|---|---|---|
| 0 | ...............1 | Standard input device |
| 1 | ..............1. | Standard output (character) device |
| 2 | .............1.. | NUL device |
| 3 | ............1... | Clock device |
| 4 | ...........1.... | Driver services interrupt 29 |
| 5 | ..........1..... | Reserved |
| 6 | .........1...... | Driver supports generic IOCTL (I/O Control) |
| 7 | ........1....... | Reserved |
| 8 | .......1........ | Reserved |
| 9 | ......1......... | Reserved |
| A | .....1.......... | Reserved |
| B | ....1........... | Open/close/removable media not supported |
| C | ...1............ | Reserved |
| D | ..1............. | Either a character device that does not support output-till-busy operation or a block device that does not use IBM block format |
| E | .1.............. | IOCTL supported |
| F | 1............... | Character device — block device when empty |

## Summary

This chapter has examined many hardware factors related to configuring and using add-on devices. It has examined the following key points:

- The microprocessor tasks are governed by interrupts, and the interrupts are classified as hardware and software. When an interrupt occurs, the microprocessor executes an interrupt service routine whose address is stored in the interrupt vector table.
- All hardware interrupts are managed by the programmable interrupt controller, which is connected to the hardware devices by way of IRQ lines. Normally, every separate device must have a unique IRQ line.
- Memory moves can by expedited by using the DMA (direct memory access) controller chip. DMA moves theoretically can be executed along multiple channels, but usually only backup software uses more than one channel at a time.
- The expansion bus is used to connect add-on devices to the microprocessor and related circuitry. The primary bus types evident in PCs are ISA, MCA, and EISA. ISA has been given fresh life, thanks to local bus technology.
- Device drivers act as translators that enable foreign devices to communicate with DOS and the microprocessor. You can review installed device drivers with the MSD and MEM commands.

DOS 6 introduced a set of new commands to support multiple boot configurations. Chapter 8 describes how to use these commands to create start-up menus and how to identify common configuration commands.

## Chapter 8

# Power System Configuration Techniques

**In This Chapter**

- Ways to interrupt and control the boot cycle
- How to create multiboot configurations
- A CONFIG.SYS checklist
- An AUTOEXEC.BAT checklist

Many of today's PCs are packed full of add-on goodies such as fax cards, scanners, CD-ROMS, sound cards and the like. Unfortunately, most of these add-ons need special device drivers or programs, which in turn reduce the amount of available memory left to run applications programs.

Several enhancements in DOS 6 enable you to precisely control which programs and device drivers are loaded when your system boots. If you don't use the CD-ROM every session, for example, you can easily configure the system to prompt for confirmation before installing the CD-ROM device driver. This chapter explains how to take advantage of all the new DOS 6 configuration options and provides a checklist of the most beneficial CONFIG.SYS and AUTOEXEC.BAT commands.

## Interrupting and Controlling the Boot Cycle

Before DOS 6, the most common way to halt the boot process was to thump the Ctrl-Break key repeatedly until DOS, somewhere in AUTOEXEC.BAT, responded with the prompt

```
Terminate batch job (Y/N)?
```

If a rogue statement in CONFIG.SYS caused the system to hang, about the only solution was to boot from a floppy and then edit the offending CONFIG.SYS statement(s).

Thankfully, DOS 6 provides a few more graceful and effective ways to interrupt and control the commands executed during the system boot.

## Using the clean-start key

In the initial boot phase, before CONFIG.SYS entries are processed, DOS displays the following message by default:

```
Starting MS-DOS...
```

When this message appears, you have a couple of seconds to press and release the F5 function key. When you press F5, the system ignores all commands in CONFIG.SYS and AUTOEXEC.BAT and performs a *clean boot*; that is, the system boots without processing any of the commands in the CONFIG.SYS or AUTOEXEC.BAT files. This feature saves you from having to create (and later find) a DOS 6 boot diskette. And it makes you wonder why it wasn't implemented a long time ago. By the way, if you hold the F5 key down too long, @ characters will cascade down the screen until you release the key.

If you are a little heavy-handed, as I am, or get tense waiting for the right moment to press and release F5, I have good news. Instead of waiting poised over the F5 key, you can just hold down one of the Shift keys. DOS clean boots just as if you pressed F5, and you don't get all the @ symbols spilling down the display because you held the F5 key down too long.

Better still, if you configured your system with a start-up menu, you can just press F5 when the menu is displayed. DOS ignores the menu options and clean boots.

Be aware that *none* of your CONFIG.SYS and AUTOEXEC.BAT commands are executed when you clean boot the system. No memory manager, no COMSPEC, no PATH, no PROMPT — as my sister would say, "No nothin'." If you need to make some corrections to the boot files, start by entering something similar to the following two commands:

```
prompt $p$g
path c:\dos
```

In addition, if COMMAND.COM is not located in the root directory, enter the following command (specifying the appropriate path to COMMAND.COM):

```
set comspec=c:\dos\command.com
```

You should consider creating a multiboot menu (discussed later in the chapter) that includes a *Clean Start* item that executes these basic commands.

If your drive C is a DoubleSpace volume, the system still boots successfully using the clean-boot technique. The device driver responsible for mounting the DoubleSpace volume is installed (by IO.SYS) before the commands in CONFIG.SYS are processed.

If you are using a third-party disk compression product, such as Stacker, however, the clean-boot technique bypasses the driver responsible for mounting your compressed volume; you are left looking at your host drive instead of the one containing the compressed files. If you're using a third-party compression product, you should consider creating a start-up menu that includes a Clean Start item that loads the disk compression device driver and nothing else. Start-up menus are thoroughly discussed later in the chapter.

## Using the interactive start key

Although the clean-start key is useful, more often than not you want to execute some, but not all, commands. For example, you may want to load only HIMEM.SYS, EMM386.EXE, and MOUSE.SYS. In such circumstances, press the F8 key as soon as the boot message `Starting MS-DOS...` is displayed. DOS prompts for confirmation before executing each command in CONFIG.SYS. If the first line of CONFIG.SYS loads HIMEM.SYS, DOS prompts you as follows:

```
device=c:\dos\himem.sys [Y,N]?
```

If you press Y, the statement executes; if you press N, the statement is bypassed. DOS then proceeds to the next statement in CONFIG.SYS and prompts you for confirmation before executing it. The process is repeated for every statement in CONFIG.SYS.

When all the CONFIG.SYS commands are processed, DOS prompts you with the following message:

```
Process AUTOEXEC.BAT [Y,N]?
```

If you press Y, AUTOEXEC.BAT is executed as normal. If you press N, the entire AUTOEXEC.BAT file is bypassed. Unfortunately, you cannot instruct DOS to individually step through each command in AUTOEXEC.BAT. Maybe next time.

If your system is configured with a start-up menu, the F8 key behaves differently. When the start-up menu is presented, the status of the F8 key is displayed at the bottom of the display as follows:

```
F8=Confirm each CONFIG.SYS line [N]
```

If you press F8, the status changes to [Y]; pressing the F8 key toggles the setting between [Y] and [N]. After you set the F8 switch as desired, you must then choose a menu item before proceeding. After all, DOS needs to know which configuration commands to step through.

If you step through some commands and want to exit without processing any more CONFIG.SYS commands, a shortcut is to press the F5 key while DOS is prompting for [Y,N]. DOS ignores the current command, all subsequent commands, and AUTOEXEC.BAT and drops you to the DOS prompt without further ado.

Similarly, if you are stepping through the CONFIG.SYS commands and want to execute all remaining commands along with AUTOEXEC.BAT, just press Esc. DOS stops prompting you and executes all remaining commands in one fell swoop.

If your system will not boot, the F8 key provides a very simple debugging tool. By stepping through the individual commands, you can ascertain precisely when the system locks up. The last command executed before lockup may be causing the problem. Try booting a second time using F8 and press N when you are prompted to confirm the suspect statement. If everything appears to boot normally, you have a good idea about the cause of the problem.

## Forcing a [Y/N] prompt

If you only want to load a specific command in CONFIG.SYS occasionally, you can force a [Y/N] confirmation prompt by inserting a question mark character (?) before the equal sign (=) in the statements. This instructs DOS to prompt for confirmation before executing the statement. The ? character is supported by all CONFIG.SYS commands except SWITCHES.

For example, you can use the ? command with the device driver for a hand-held scanner. You don't use the scanner everyday, so you don't want to waste memory loading the device driver. But you also don't want to edit CONFIG.SYS and reboot every time you want to load the scanner. If you add a ? to the device statement, DOS prompts you to confirm execution of the device driver every time you boot the system. Here's an example of how to write such a statement:

```
device?=c:\scanman\hhscand.sys /a=280 /i=1
```

If you press Esc when prompted to confirm execution of a device installed with the ? option, DOS assumes that your response to the prompt is Y. It also assumes that your response to all remaining prompts is Y and bypasses them.

## Controlling start keys with the SWITCHES command

If you are responsible for configuring computers for other users, you may be concerned that they will be able to press F5 (or Shift) and F8 to bypass the start-up commands, possibly causing mayhem. Fear not. By using the SWITCHES statement in CONFIG.SYS, you can enable or disable the start-up interruption keys.

The SWITCHES command serves a variety of purposes, and the full syntax is explained in Part IV, "The Complete DOS 6 Command Reference." However, two switches are directly related to the booting process:

/F Instructs DOS to boot immediately, without pausing for the normal two seconds when the message `Starting MS-DOS...` is displayed. If drive C is a DoubleSpace volume, there is still a perceptible delay as DOS mounts the DoubleSpace volume — it is just two seconds shorter.

/N Instructs DOS to ignore the F5 (and Shift) and F8 interruption keys. The user cannot bypass CONFIG.SYS or step through each command individually.

Normally, if you use the /N command, you should also use /F because there is no reason to delay the boot for two seconds. The following is an example CONFIG.SYS that cannot be interrupted:

```
SWITCHES=/F/N
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM I=B000-B7FF WIN=B300-B7FF
BUFFERS=15,0
FILES=40
DOS=UMB
LASTDRIVE=M
```

You do not have to locate the SWITCHES statement at the beginning of CONFIG.SYS. DOS is smart enough to peruse the entire file looking for a SWITCHES statement. In fact, if you are trying to stop users from tampering with the configuration setup, you should consider burying the SWITCHES statement in the body of the main commands. The casual user is much less likely to find and modify it.

## Creating Multiboot Configurations

A great new feature of DOS 6 is that it gives you the ability to define multiple start-up configurations in CONFIG.SYS. DOS can display a menu of start-up options and then only execute the commands defined for the option you select. For example, the following menu is displayed on my notebook system:

```
MS-DOS 6 Startup Menu
=====================

  1. Clean
  2. InterLink
  3. Normal
```

You define start-up menus in CONFIG.SYS using the commands in Table 8-1.

**Table 8-1 Start-Up Menu Commands**

| *Command* | *Description* |
|---|---|
| [MENU] | Identifies the beginning of a series of menu items and commands that define the start-up menu. |
| INCLUDE | Includes the commands from one configuration block in another. |
| MENUCOLOR | Sets the foreground and background display colors for the menu. |
| MENUDEFAULT | Sets the default item in a menu and (optionally) the number of seconds to wait for user input before applying the default. |
| MENUITEM | Creates an item on the menu and identifies the name of the associated configuration block. When the user selects the menu item, DOS executes the commands in the specified configuration block. The order of the MENUITEM commands in CONFIG.SYS defines the order of the menu options. |
| NUMLOCK | Because the user can select a menu item by pressing the menu number, this command was added to provide a convenient way to turn off Num Lock. |
| SUBMENU | Creates a branch to a secondary menu, thereby providing nested or multilevel configuration menus. |
| [*BLOCKNAME*] | Every block specified with the MENUITEM command must have a corresponding block defined. This approach is similar to the batch file command GOTO, where the GOTO name must have a corresponding label elsewhere in the file. |

## Defining menus

The following CONFIG.SYS file, which was used to create the three-option menu presented in the preceding section, provides a complete example of how to create start-up menus:

```
[menu]
menuitem=Clean
menuitem=InterLink
menuitem=Normal
numlock=off

[Clean]
BUFFERS=15,0
FILES=30
LASTDRIVE=G
FCBS=4,0

[Normal]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB
```

```
BUFFERS=15,0
FILES=30
LASTDRIVE=G
FCBS=4,0
SHELL=C:\DOS\COMMAND.COM C:\DOS\  /p
DEVICEHIGH /L:1,47296 =C:\DOS\DBLSPACE.SYS

[InterLink]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB
DEVICE=C:\DOS\INTERLNK.EXE /DRIVES:5
BUFFERS=15,0
FILES=30
LASTDRIVE=G
```

The anatomy of a multiconfiguration CONFIG.SYS file is straightforward. One [MENU] section includes a set of MENUITEM statements that define each menu option. The body of CONFIG.SYS contains configuration blocks, with each *blockname* identified by the MENUITEM name and enclosed in square brackets.

When the user selects a menu item, DOS proceeds to the appropriate configuration-block heading — for example, [InterLink] — and executes the commands that follow. DOS executes all the commands up to the next configuration-block heading or the end of the file, whichever comes first.

A blockname can be up to 70 characters long but must not include embedded spaces or any of the following characters:

\ / [ ] = ; ,

Note that any spaces preceding or following the blockname are ignored.

DOS displays a maximum of nine entries in any single menu.

NOTE

If a menu item is clearly listed in CONFIG.SYS but does not show up in the start-up menu, it's likely that the problem is related to the [*blockname*] entry. If DOS cannot find a menu item's corresponding configuration block, the MENUITEM statement is ignored. Check that the blockname specified with the MENUITEM statement is the same as the blockname specified in the body of CONFIG.SYS. Blocknames are not case sensitive.

Specify an optional MENUITEM title when you don't want the single word *blockname* to be used as the title. To specify a more descriptive MENUITEM title, use the following syntax:

```
menuitem=blockname[,menudescription]
```

You could enhance the [MENU] section in the example as follows:

```
[menu]
menuitem=Clean,Plain boot with nothing loaded
menuitem=InterLink,Standard Setup with InterLink loaded
menuitem=Normal,Normal Configuration Settings
numlock=off
```

After you make these enhancements, DOS displays the start-up menu at system boot as follows:

```
MS-DOS 6 Startup Menu
=====================

   1. Plain boot with nothing loaded
   2. Standard Setup with InterLink loaded
   3. Normal Configuration Settings
```

## Setting menu defaults

The MENUDEFAULT command controls the default menu item — that is, the item that is highlighted when the menu is first displayed — and optionally specifies a timeout period. If the user does not press a key within the timeout period, DOS proceeds as though the default item was selected. The timeout period can range from 0 to 90 seconds. Locate the MENUDEFAULT statement in the [MENU] section and use the following syntax:

menudefault=*blockname*[,*timeout*]

The following [MENU] section of CONFIG.SYS defines the third item in the example menu as the default and assigns a timeout period of 10 seconds:

```
[menu]
menuitem=Clean,Plain boot with nothing loaded
menuitem=InterLink,Standard Setup with InterLink loaded
menuitem=Normal,Normal Configuration Settings
numlock=off
menudefault=Normal,10
```

If the MENUDEFAULT command is not used, the first menu item is highlighted as the default, and DOS waits for the user to make a selection and press the Enter key.

## Changing menu colors

DOS has never been great with colors, and the new start-up menu facility is no exception. But you can at least change the basic display colors. You can modify the standard foreground and background menu colors by adding a MENUCOLOR statement to the [MENU] section. Use the following syntax:

menucolor=*f*[,*b*]

**Table 8-2 Color Codes for the MENUCOLOR Command**

| Code | Color | Code | Color |
|---|---|---|---|
| 0 | Black | 8 | Dark Gray |
| 1 | Blue | 9 | Light Blue |
| 2 | Green | 10 | Light Green |
| 3 | Cyan | 11 | Light Cyan |
| 4 | Red | 12 | Light Red |
| 5 | Magenta | 13 | Light Magenta |
| 6 | Brown | 14 | Yellow |
| 7 | Light Gray | 15 | White |

The *f* variable is the foreground text color code, and *b* is the background color code. The color code must be a value in the range 0 to 15. Table 8-2 shows the codes along with the corresponding colors.

Some systems do not support background colors in the range 8 through 15. If you get blinking or flashing text, change the background color to a code in the range 0 to 7. Unless, of course, you want blinking text!

**NOTE**

Unfortunately, you cannot change the heading colors or the color of the highlight bar. The highlight bar is always light gray, so avoid using a light gray background (code 7), or the highlight bar will not be visible.

The color settings apply to the output generated by AUTOEXEC.BAT commands and other DOS commands. Add a `cls` command at the beginning of the AUTOEXEC.BAT file to clear the display and set the colors back to the default light gray on black.

### Changing menu headings

Bad news: There is no easy way to change the bland `MS-DOS 6 Startup Menu` menu heading. The menu heading is a constant in the hidden IO.SYS file. Although you can hack around in IO.SYS and change the title, I don't recommend it.

## Combining common commands

If every configuration block shares one or more configuration statements, you don't need to repeat the commands in every block. The special configuration block called [COMMON] provides a useful shortcut. All commands included in a [COMMON] block are executed no matter which menu item the user selects.

In the example CONFIG.SYS file, all three blocks included the following commands:

```
BUFFERS=15,0
FILES=30
LASTDRIVE=G
```

You can simplify the entire CONFIG.SYS file by adding the commands to a common section, as follows:

```
[menu]
menuitem=Clean
menuitem=InterLink
menuitem=Normal
numlock=off

[common]
BUFFERS=15,0
FILES=30
LASTDRIVE=G

[Clean]
FCBS=4,0

[Normal]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB
FCBS=4,0
SHELL=C:\DOS\COMMAND.COM C:\DOS\  /p
DEVICEHIGH /L:1,47296 =C:\DOS\DBLSPACE.SYS

[InterLink]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB
DEVICE=C:\DOS\INTERLNK.EXE /DRIVES:5
```

You can place more than one [COMMON] block in a CONFIG.SYS file. All commands situated in common blocks are processed *after* the commands for the block selected by the user. Multiple common blocks are executed in the order they are entered in CONFIG.SYS.

One disadvantage of the [COMMON] block is that its commands are always executed. If you have a menu option for a clean boot configuration, avoid using the [COMMON] feature. After all, it wouldn't be a clean boot if all the common commands were executed.

If you add commands to the beginning of CONFIG.SYS, prior to the first MENU or [*blockname*] statement, the commands will be assumed to be part of a COMMON block and so will be executed no matter which menu option is selected.

If you have one or more commands that are shared by some but not all menu items, you can use the INCLUDE command to incorporate the shared commands in selective blocks. The syntax of the include statement is as follows:

```
include=blockname
```

When DOS encounters an INCLUDE statement, processing branches to the specified block. When all the commands in the block are processed, DOS returns to the statement on the line following the INCLUDE statement.

In the example configuration, the [Normal] and [Interlink] sections both load HIMEM.SYS and EMM386. The file can be modified to include a new section, [MEMSTUFF], as follows:

```
[menu]
menuitem=Clean
menuitem=InterLink
menuitem=Normal
numlock=off

[common]
BUFFERS=15,0
FILES=30
LASTDRIVE=G

[memstuff]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM

[Clean]
FCBS=4,0

[Normal]
INCLUDE=MEMSTUFF
DOS=HIGH,UMB
FCBS=4,0
SHELL=C:\DOS\COMMAND.COM C:\DOS\  /p
DEVICEHIGH /L:1,47296 =C:\DOS\DBLSPACE.SYS

[InterLink]
INCLUDE=MEMSTUFF
DOS=HIGH,UMB
DEVICE=C:\DOS\INTERLNK.EXE /DRIVES:5
```

Many software installation programs make modifications to CONFIG.SYS, and few of them (as yet) are aware of DOS 6's multiconfiguration facility. Usually, the installation programs add their commands to the end of the file. Consider adding a [COMMON] statement to the end of CONFIG.SYS. With this statement, CONFIG.SYS is better suited to accommodate ad-hoc commands added to the end of CONFIG.SYS by installation software. If an installation program subsequently adds commands to the end of CONFIG.SYS, they will be situated in this COMMON section; therefore, the commands will be executed no matter which item the user selects from the start-up menu.

Better still, if one menu item is used as the "business as usual" option, add a custom blockname statement, say [DEFAULT], to the end of CONFIG.SYS and then use the INCLUDE menu command to insert the following statement at the end of the business-as-usual block:

```
include=default
```

Whenever the user selects the business-as-usual option from the start-up menu, the commands situated at the end of CONFIG.SYS (after the [DEFAULT] line) will automatically be executed.

It is not always beneficial to place common commands in a [COMMON] block or to share them with an INCLUDE statement, especially if you are optimizing your system for memory usage. The load order for device drivers and the load position in an upper memory region probably differ from configuration to configuration. In one configuration, you may load MOUSE.SYS in upper memory with DEVICEHIGH, but in another configuration, when other, larger programs may make better use of upper memory, you may load MOUSE.SYS low with a DEVICE statement.

Furthermore, MEMMAKER (the DOS 6 memory-optimization program) is not very adept at dealing with CONFIG.SYS files that contain menu blocks. Refer to Chapter 5 for more information on memory optimization and how to optimize multiconfiguration systems.

## Using multimenu configurations

Because there is a limit of nine items in each menu, you may want to use the SUBMENU command to create submenus. Even if you don't fill all nine item slots, you can use SUBMENU to organize your menu system into a hierarchy.

By using the SUBMENU command in place of MENUITEM, you instruct DOS to display another menu rather than execute standard CONFIG.SYS commands. When the user selects a menu item defined with SUBMENU, a secondary menu is automatically displayed. The syntax of the SUBMENU command is as follows:

```
submenu=blockname[,menudescription]
```

The block specified by the *blockname* argument is a configuration block that defines another menu. The submenu configuration block can include MENUITEM, MENUCOLOR, MENUDEFAULT, NUMLOCK, and even further SUBMENU commands. You can nest menus at least ten levels deep, but if you nest menus more than two or three levels, you probably are in need of counseling.

The following CONFIG.SYS file defines a two-menu system (with commands related to the submenu in bold):

```
[menu]
menuitem=Clean
menuitem=InterLink
menuitem=Normal
submenu=Graphics,Publishing Menu
menucolor=14,2
numlock=off

[common]
BUFFERS=15,0
FILES=30
LASTDRIVE=G

[memstuff]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM

[Clean]
FCBS=4,0

[Normal]
INCLUDE=MEMSTUFF
DOS=HIGH,UMB
FCBS=4,0
SHELL=C:\DOS\COMMAND.COM C:\DOS\  /p
DEVICEHIGH /L:1,47296 =C:\DOS\DBLSPACE.SYS

[InterLink]
INCLUDE=MEMSTUFF
DOS=HIGH,UMB
DEVICE=C:\DOS\INTERLNK.EXE /DRIVES:5

[Graphics]
menuitem=Scanner,Install Scanner
menuitem=Tablet,Install Digitizing Tablet
menuitem=ScanTab,Install Scanner and Tablet
submenu=menu,Return to Main Menu
menucolor=15,4
```

```
[Scanner]
DEVICE=C:\SCANMAN\HHSCAND.SYS /A=280 /I=1

[Tablet]
DEVICE=C:\BUSTER\TABSCMD.SYS /P /W=54 /T=3

[ScanTab]
INCLUDE=SCANNER
INCLUDE=TABLET

[common]
```

When the system boots, DOS displays the following menu:

```
MS-DOS 6 Startup Menu
=====================


  1. Clean
  2. InterLink
  3. Normal
  4. Publishing Menu
```

If the user selects option 4, the following submenu is displayed in a different color:

```
MS-DOS 6 Startup Menu
=====================


  1. Install Scanner
  2. Install Digitizing Tablet
  3. Install Scanner and Tablet
  4. Return to Main Menu
```

## Providing an escape option

Unlike the majority of menu systems, DOS, for some odd reason, does not allow you to escape from a submenu back to the main menu. Once you choose an option that displays a submenu, you are stuck there.

UNDOCUMENTED FEATURE

If you create a boot menu system that includes submenus, always add a final menu option called *Return to Main Menu.* Although it is not documented, if you specify *menu* as the blockname, DOS jumps back to the initial menu. To provide the user with an easy way to return to the main menu, just add the following statement to the end of each submenu definition:

```
submenu=menu,Return to Main Menu
```

## Prompting for user confirmation

You can use the submenu facility in an unconventional way: to instruct DOS to prompt the user for confirmation before proceeding with the selected option. The approach is best explained by example.

When setting up a menu system, you may want to give the user a chance to tell DOS to abort before it executes a particular configuration. Suppose, for example, that you want to remind users that they must attach a scanner to the system and turn it on before installing the scanner's device driver.

You can instruct DOS to display a submenu with the two options Yes and No, prompting the user for confirmation before it proceeds. The following CONFIG.SYS file defines a menu with two options, Normal and Scanner:

```
[menu]
menuitem=Normal
submenu=Scanner,Activate Scanner
menucolor=14,2
numlock=off

[normal]
BUFFERS=15,0
FILES=30
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB

[Scanner]
menuitem=LoadIt,Yes: The scanner is connected and turned on
submenu=menu,No:  I forgot to get the scanner. Abort
scanner load
menucolor=15,4

[LoadIt]
include=normal
DEVICE=C:\SCANMAN\HHSCAND.SYS /A=280 /I=1

[common]
```

When the system boots, the following menu is displayed:

```
MS-DOS 6 Startup Menu
=======================

   1. Normal
   2. Activate Scanner
```

If the user selects the Scanner option, the following confirmation menu is displayed:

```
MS-DOS 6 Startup Menu
=====================

      1. Yes: The scanner is connected and turned on
      2. No:  I forgot to get the scanner. Abort scanner load
```

When the user selects Yes, the scanner is installed and the system boots. When the user selects No, the main menu is displayed again.

***Note***: The program does not limit the number of menus and submenus that you can branch to.

## Configuring AUTOEXEC.BAT

As yet, DOS does not offer a built-in feature that enables you to display a full-screen menu from AUTOEXEC.BAT. However, you can edit AUTOEXEC.BAT so that it determines which item the user selected from the CONFIG.SYS menu and then executes the appropriate batch file statements.

When a user makes a choice from a start-up menu, DOS creates an environment variable to indicate which item the user selected. The name of the environment variable is CONFIG, and the value is the selected blockname. For example, if the user chose the Normal option in the previous example, DOS would create the following environment variable:

```
CONFIG=Normal
```

A batch file can evaluate the value of the environment variable by using the expression %***varname***%. To test the value of the CONFIG environment variable, use the syntax `%config%`. After executing some standard AUTOEXEC.BAT statements, such as PATH and PROMPT, you can test the value of CONFIG with an IF statement. Here is an example:

```
path c:\dos;c:\windows;c:\batfiles
prompt $p$g
set temp=d:\temp
c:\dos\doskey
if not "%config%"=="Scanner" goto quit
c:\scanman\scanstrt /l
:quit
```

If you have several menu options, you might consider using a GOTO statement rather than several IF statements. By adding a GOTO %CONFIG% statement, you can branch to a specific label in the batch file, based on the user's menu selection. The following batch file illustrates this approach:

```
path c:\dos;c:\batfiles
prompt $p$g
goto %config%
:Interlnk
echo Execute InterSvr on Desktop to start Interlnk
goto quit
:Normal
c:\dos\doskey
goto quit
:Windows
path=c:\windows;%path%
set temp=c:\wintemp
win
goto quit
:Clean
echo Maximum Memory Configuration Installed
:Quit
```

If the message `Label not found` is sometimes displayed when you boot the system, it is caused, not surprisingly, by a missing label. Check and make sure that you have one label for every possible menu selection in CONFIG.SYS.

Because DOS automatically creates a CONFIG environment entry, you might consider freeing some environment space by adding the following command to the end of AUTOEXEC.BAT:

```
set config=
```

You can take this approach to extremes and use the multimenu facility in CONFIG.SYS solely as a front-end menu for AUTOEXEC.BAT. The following CONFIG.SYS file displays a menu, but all the executable commands are stored in the COMMON section:

```
[menu]
menuitem=Mouse,Use the standard mouse
menuitem=Tablet,Use the digitizing tablet
menuitem=Trackball,Use the trackball
numlock=off

[common]
BUFFERS=15,0
FILES=30
```

```
LASTDRIVE=G
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB
FCBS=4,0
SHELL=C:\DOS\COMMAND.COM C:\DOS\  /p
DEVICEHIGH /L:1,47296 =C:\DOS\DBLSPACE.SYS
[Mouse]
[Tablet]
[Trackball]
[common]
```

No matter which menu item the user chooses, the same configuration commands are executed. The reason for making the user choose an option is so that specific AUTOEXEC.BAT commands can be executed based on the value of %CONFIG%, as follows:

```
path c:\dos;c:\batfiles
prompt $p$g
goto %config%
:Mouse
c:\dos\mouse
goto quit
:Tablet
c:\tb\starttb /x=640 /y=480
goto quit
:Trackball
c:\ltech\tball
:quit
```

Don't forget that you can use the CHOOSE batch command to generate single-line prompts that request user input. For more information about CHOOSE, IF, and other batch file techniques, refer to Chapter 10.

## Checking CONFIG.SYS

The precise entries in CONFIG.SYS vary from system to system. Apart from the menu-related commands, CONFIG.SYS supports nearly 20 other configuration commands. Table 8-3 provides a summary of the primary configuration commands. Use it to verify that you have addressed each area in your system's configuration.

| **Table 8-3** | **The Most Common CONFIG.SYS Entries** | |
|---|---|---|
| ***Category*** | ***Command*** | ***Description*** |
| Devices | DEVICE | Loads a device driver. Usually, the device driver is loaded into base memory, but some device drivers, such as INTERLNK, automatically load into upper, expanded, or extended memory. |
| | DEVICEHIGH | Loads a device driver into upper memory. If there isn't enough room or if no upper memory is available, the device is loaded in base memory. |
| Memory | DOS | Use with the HIGH (on 286+ systems) and UMB (on 386+ systems) switches to load DOS into high memory and to instruct DOS to manage the upper memory area, respectively. |
| | HIMEM.SYS | On 286+ systems, load HIMEM.SYS, using DEVICE, to manage extended memory and provide access to the high memory area (HMA). |
| | EMM386.EXE | Provides advanced memory management capabilities on 386+ systems. |
| | INSTALL | Loads memory-resident programs from CONFIG.SYS and avoids allocating unnecessary environment space for them. |
| | FCBS | Specifies the maximum number of file control blocks DOS can have open at any given time. Only very old programs use file control blocks. To minimize the FCBS allocation and save memory, use the statement `FCBS=1`. |
| | DBLSPACE.SYS | Transfers the DoubleSpace device driver into upper memory. |
| | STACKS | Allocates memory to manage hardware interrupts. If you encounter stack error messages, such as `Internal Stack Overflow`, you may need to add a STACKS statement to CONFIG.SYS or increase the number and size of the stack frames. |
| Files/drives | FILES | Specifies the number of files DOS can have open concurrently. Use a value in the range 20 to 50. |
| | BUFFERS | Specifies the number of disk buffers used by DOS when reading files. If you use a disk cache program (such as SMARTDRV), set the buffers to a small value, such as 4. Otherwise, use a value in the 20 to 40 range, depending on memory requirements. |
| | LASTDRIVE | Controls how many logical drives are recognized by DOS. Do not assign drives you don't need; it wastes base memory. |
| | RAMDRIVE.SYS | Creates a virtual hard drive using base, extended, or expanded memory. |

*(continued)*

**Table 8-3 The Most Common CONFIG.SYS Entries *(continued)***

| *Category* | *Command* | *Description* |
|---|---|---|
| International | COUNTRY.SYS | Configures the country language conventions for your system. Areas affected by COUNTRY.SYS are the date and time display, the characters of the alphabet, the symbol used to denote currency, and the sort order of files. If you do not have a COUNTRY entry in CONFIG.SYS, your system is automatically configured as a U.S.A. system. |
|  | DISPLAY.SYS | Provides international character support by allowing code-page switching. You only need to use this device driver if you configure your system for international use. |
| Miscellaneous | ANSI.SYS | A device driver that provides ways to improve DOS display capabilities, control the cursor location, and assign special functions to keys. |
|  | BREAK | Enables or disables checking for Ctrl-Break during disk access and other nondisplay activity. |
|  | SHELL | Sets the options for COMMAND.COM or allows a different command processor to be loaded. SHELL is most often used with the /E switch to control the memory allocated for the environment. |
|  | SWITCHES | Controls four configuration switches for customizing the behavior of the operating system. |

As a rule of thumb, your CONFIG.SYS should include at least the following entries:

HIMEM.SYS (on 286+ systems)
EMM386.EXE (on 386+ systems)
DOS=HIGH,UMB (on 286+, 386+ systems)
FILES
FCBS
BUFFERS
LASTDRIVE
SHELL

You probably will also have a collection of DEVICE and DEVICEHIGH statements that load other device drivers, such as MOUSE.SYS. The following is a typical CONFIG.SYS file from a 386 system that uses Workgroup Connection:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM I=B000-B7FF WIN=B300-B7FF
BUFFERS=15,0
FILES=40
DOS=HIGH,UMB
```

```
LASTDRIVE=M
FCBS=1
STACKS=9,256
;Set Environment size
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /e:431 /p
;Install network drivers
DEVICEHIGH /L:2,55168 =C:\WINDOWS\MOUSE.SYS /Y
DEVICE=C:\WINDOWS\PROTMAN.DOS /i:C:\WINDOWS
DEVICEHIGH /L:2,11168 =C:\WINDOWS\EXP16.DOS
DEVICEHIGH /L:2,7280 =C:\WINDOWS\WORKGRP.SYS
DEVICEHIGH=C:\DOS\ANSI.SYS
```

Remember that DOS ignores any lines beginning with REM or the semicolon character (;). With the introduction of the menu commands, CONFIG.SYS is becoming as complicated and as intricate as batch files. Take advantage of REM and ; to add explanatory comments to CONFIG.SYS.

If you modify your configuration file, be sure to use MEMMAKER to make optimum use of memory. Refer to Chapter 5 for a full discussion of memory optimization and its impact on CONFIG.SYS.

## Checking AUTOEXEC.BAT

There are almost as many different AUTOEXEC.BAT configurations as there are PCs. Nonetheless, some standard commands should be used in most cases. Table 8-4 lists some of the common AUTOEXEC.BAT entries you should consider utilizing.

**Table 8-4 Common AUTOEXEC.BAT Entries**

| *Command* | *Description* |
|---|---|
| PATH | Stores, in the environment, a list of directories that DOS searches when looking for a program or batch file. Specify the most commonly accessed directories first. |
| PROMPT | Defines, in the environment, the default command prompt. The most common prompt argument is $p$g, which represents the default drive and directory. |
| MOUSE | If your mouse is not installed as a device driver, execute MOUSE.COM. |
| SET TEMP= | More and more programs are referencing a TEMP environment variable to determine where to store temporary files. Specify a drive and directory, and if you have more than one hard disk, specify a directory on the fastest disk. |
| DOSKEY | Installs the DOS keyboard macro utility. |
| SMARTDRV | Configures and starts the DOS disk cache program. Use this if you aren't using another third-party disk cache. |
| MIRROR | Use the DOS 5 command MIRROR to make a backup copy of the hard disk partition table and record UNFORMAT details for each hard drive. |

## Summary

To get the most from your system, you must optimally configure CONFIG.SYS and AUTOEXEC.BAT. In this chapter, you examined the following points:

- You can *clean boot* the system (bypassing CONFIG.SYS and AUTOEXEC.BAT) by pressing F5 when DOS displays the message `Starting MS-DOS...` or by holding down the Shift key.
- You can selectively execute commands by pressing F8 when DOS starts or by placing a question mark (?) before the equals character (=) in a configuration command.
- If you want to ensure that all CONFIG.SYS commands are executed without user intervention, you can use the SWITCHES command to disable the F5 and F8 keys.
- You can use the new menu commands to create start-up menus. Each menu can contain as many as nine items, and each menu can branch to another menu. You should consider adding a *Return to Main Menu* option to each submenu.
- Although AUTOEXEC.BAT menus are not directly supported, the CONFIG environment variable can be tested to determine which menu item the user selected.
- Configuration files differ from system to system, but the majority of systems use a common set of commands. By looking at typical configuration files, you can determine whether anything obvious is missing from your system configuration.
- If you change CONFIG.SYS and AUTOEXEC.BAT, re-run MEMMAKER to optimize the new configuration.

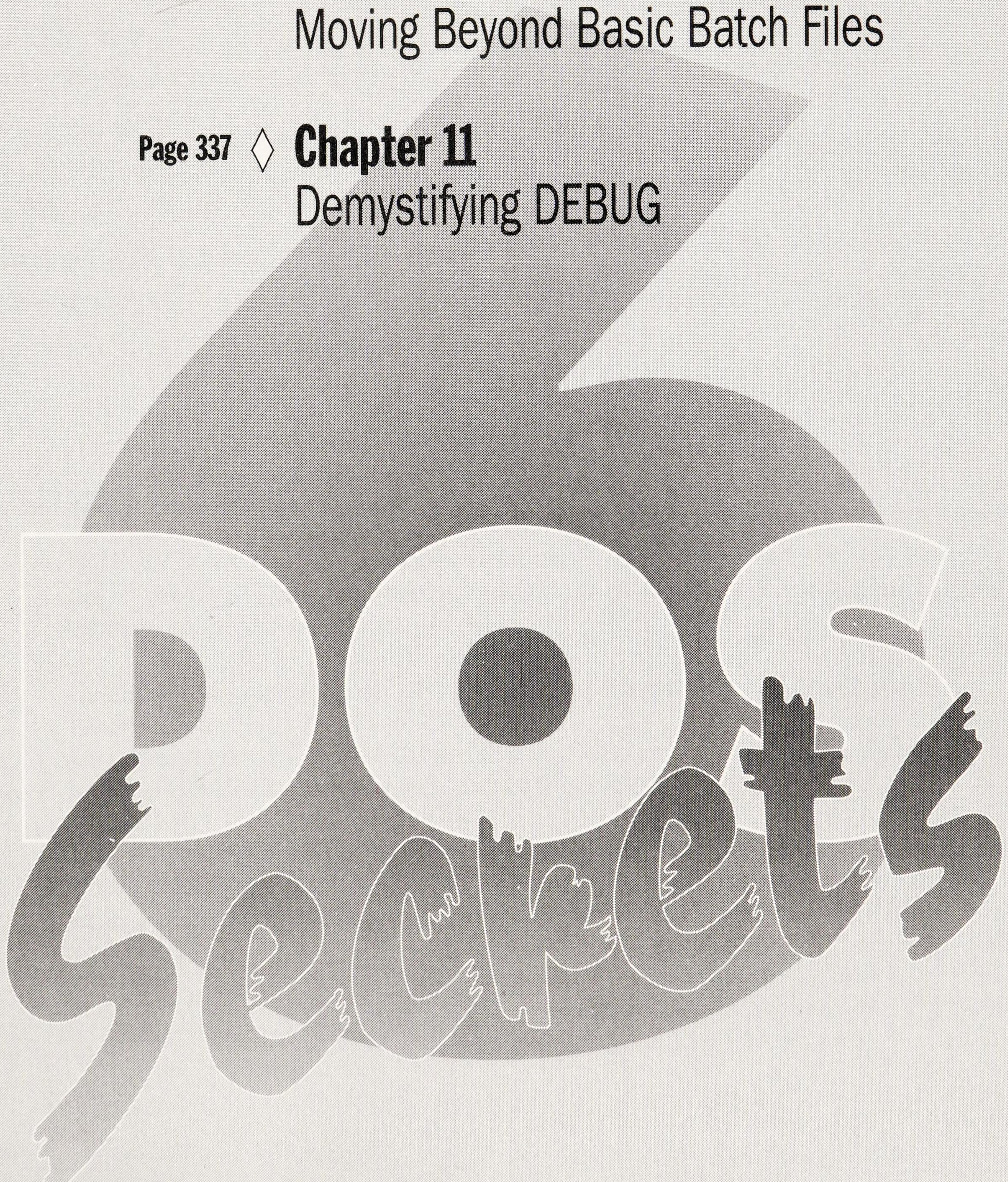
Now you are ready to begin Part III of this book, in which you learn how to use DOS tools ranging from DOSKEY to DEBUG to automate your tasks and improve your productivity.

# Part III:
# Secrets for Automating DOS

Page 285 ◊ **Chapter 9**
Putting DOSKEY to Work

Page 305 ◊ **Chapter 10**
Moving Beyond Basic Batch Files

Page 337 ◊ **Chapter 11**
Demystifying DEBUG



## Chapter 9

# Putting DOSKEY to Work

### In This Chapter

- Installing DOSKEY
- Reinstalling DOSKEY
- Using DOSKEY as a command-line manager
- Using DOSKEY macros
- Comparing DOSKEY macros to batch files
- My favorite DOSKEY macro tricks
- Moving beyond DOSKEY with ANARKEY

## Installing DOSKEY

DOSKEY is a versatile tool that provides three primary and very useful functions: the ability to recall previous commands to save retyping, command-line editing (no more backspacing), and a simple and effective macro utility.

DOSKEY uses less than 4K of memory, and even that small amount can be loaded high. Because of DOSKEY's low memory demands and the features it provides, every DOS user should use DOSKEY or a similar third-party utility such as ANARKEY (discussed later in this chapter).

Installing DOSKEY couldn't be easier; you just enter the DOSKEY command with no switches, and DOSKEY is loaded as a TSR. I recommend that you add a DOSKEY statement to your AUTOEXEC.BAT file to install DOSKEY every time you boot.

## Setting Insert Mode

Command-line editing is set by default to Overtype Mode. If you prefer the default to insert characters, enter the DOSKEY command as follows:

```
doskey /insert
```

**NOTE**

You can always press the Ins key to toggle Insert Mode when you are entering an individual command. The next time you type a command, however, the default insertion mode is reinstated. After DOSKEY has been installed, you can change the default by reexecuting DOSKEY with either the /INSERT or /OVERSTRIKE switches. Believe it or not, you must type the entire switch (these long switches cannot be abbreviated). To set DOSKEY in Overtype Mode, for example, enter the following command:

```
doskey /overstrike
```

Note that this command does not change any of the other DOSKEY settings (it does not clear the command history or erase any macros).

## Setting the buffer size

DOSKEY uses a buffer to store the history of executed commands and user-defined macros. By default, 512 bytes of memory are reserved for this buffer. The /BUFSIZE switch can be used to allocate a different buffer size when you first execute DOSKEY. The following command installs DOSKEY in Insert Mode and creates a 1,024-byte buffer:

```
doskey /insert /bufsize=1024
```

The smallest buffer size is 256 bytes, and you can make it as large as you want. A buffer size of 512, 1,024, or 2,048 bytes, however, generally should meet most needs.

**ERROR MESSAGES DECODED**

If you are unable to recall old commands (by pressing the up-arrow key), DOSKEY has probably run out of buffer space. If you then try to assign a new DOSKEY macro (discussed later in this chapter), the following error message is displayed:

```
Insufficient memory to store macro. Use the DOSKEY command
with the /BUFSIZE switch to increase available memory.
```

Having seen the error message, you might be tempted to execute a command such as DOSKEY /BUFSIZE=1024, but that doesn't work either. The following error message is displayed:

```
Cannot change BUFSIZE.
```

The solution to these problems is to reinstall DOSKEY.

## Reinstalling DOSKEY

After DOSKEY has been installed, it cannot be uninstalled, but you can reinstall it by specifying new parameters. To reinstall DOSKEY, execute the DOSKEY command with the /REINSTALL switch and any other command switches, such as /BUFSIZE and /INSERT, as shown in this example:

```
doskey /reinstall /insert /bufsize=1024
```

Unfortunately, when you reinstall DOSKEY, the command history and macros are erased. Later in this chapter, in the section "Building a macro builder," you learn how to automatically create a batch file to re-create the macros.

## Using DOSKEY as a Command-Line Manager

Although DOSKEY serves a number of purposes, the vast majority of users use it for just two reasons: to edit the command line and to recall old commands. If DOSKEY served no other purposes than to edit and recall commands, it still would be a valuable tool.

### Editing the command line

Sometimes the simple things in life are the most rewarding. So it goes for DOSKEY. When you enter a command, pressing the left-arrow key moves the cursor one position to the left, but, unlike in plain DOS, the character is not erased. In other words, the left-arrow key doesn't act like the Backspace key — it works just like the left-arrow key in EDIT.

In addition to being capable of moving the cursor to the left and right one character, DOSKEY supports a variety of other editing keys (see Table 9-1).

**Table 9-1 The Editing Keys in DOSKEY**

| *Key* | *Purpose* |
|---|---|
| Left arrow | Moves the cursor one position to the left |
| Ctrl-left arrow | Moves the cursor to the beginning of the word to the left |
| Right arrow | Moves the cursor one position to the right |
| Ctrl-right arrow | Moves the cursor to the beginning of the word to the right |

*(continued)*

**Table 9-1 The Editing Keys in DOSKEY (continued)**

| *Key* | *Purpose* |
|---|---|
| Home | Moves the cursor to the beginning of the command line |
| Ctrl-Home | Erases all the characters to the left of the cursor and moves the remaining characters to the beginning of the command line |
| End | Moves the cursor to the end of the command line |
| Ctrl-End | Erases the character at the cursor and all remaining characters on the command line |
| Del | Erases the character at the cursor |
| Backspace | Erases the character to the left of the cursor |
| Esc | Erases the entire command line |
| Ins | Toggles between Insert and Overtype Modes |

## Using the command history

After DOSKEY has been loaded, every command you execute is stored in a command buffer, which forms a stack of old commands. If the DOSKEY buffer fills, the oldest commands are removed to make room for the command just entered.

When you press the up-arrow key, the last command that was executed is retyped on the command line. Press the up-arrow key a second time to display the previous command. By repeatedly pressing the up-arrow key, you can scroll through all the past commands stored in the command buffer. You can scroll forward through the command list by pressing the down-arrow key. After you have selected the command, just press Enter to execute the command or use the editing keys to modify it.

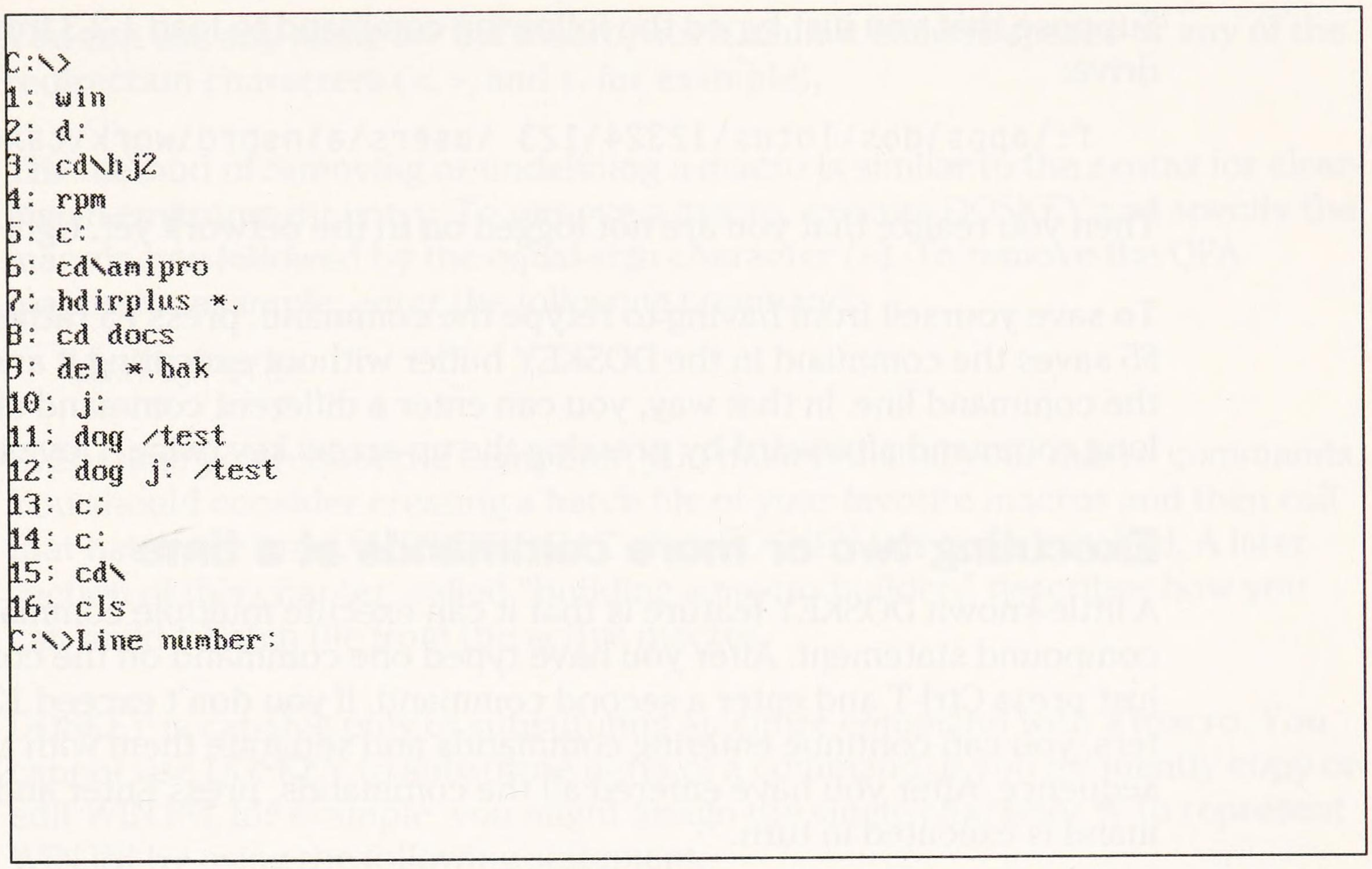
To jump to the oldest command in the buffer, press the PgUp key.

## My top five (forgotten) DOSKEY commands

The five most often used DOSKEY keys are the four arrow keys and the Ins key. If you're not using some of the other keys, you probably are not being fully productive. Here are my favorite productivity boosters:

### Selecting commands by number

When you press the F7 key, DOS lists all the commands stored in the history list, with every command prefixed by a number. The oldest command is assigned number 1, and the most recent command has the largest number. If you have



```
C:\>
1: win
2: d:
3: cd\hj2
4: rpm
5: c:
6: cd\amipro
7: hdirplus *.
8: cd docs
9: del *.bak
10: j:
11: dog /test
12: dog j: /test
13: c;
14: c:
15: cd\
16: cls
C:\>Line number:
```

**Figure 9-1:** The use of the F7 and F9 keys to list the command history and select a command by number.

more than a screen full of old commands, DOS automatically uses the MORE filter to pause the display after every screen.

After you have perused the list of old commands, you can select one by pressing F9 and then entering the command number. Figure 9-1 shows this technique being used.

### Using characters to select commands

If you can remember the first few characters of a command, you can avoid having to scroll back through all the commands to look for a specific one. Just enter one or more of the command's first characters and then press F8. DOS searches back through the command stack and, as soon as it finds a match, places the entire command on the command line.

### Saving a command without executing it

Have you ever typed a long command, prepared to press Enter, and then realized that you forgot to issue another command first? Reluctantly, you press Esc to erase the premature command and then enter the correct command. Finally, when that command has been executed, you type the long command a second time. I do it all the time.

Suppose that you just typed the following command to load 1-2-3 from a network drive:

```
f:\apps\dos\lotus\12324\123 \users\ainsbrd\work\taxes94.wk1
```

Then you realize that you are not logged on to the network yet. Ugh!

To save yourself from having to retype the command, press F5 rather than Esc. F5 saves the command in the DOSKEY buffer without executing it and then erases the command line. In that way, you can enter a different command and recall the long command afterward by pressing the up-arrow key twice. I love it!

### Executing two or more commands at a time

A little-known DOSKEY feature is that it can execute multiple commands in one compound statement. After you have typed one command on the command line, just press Ctrl-T and enter a second command. If you don't exceed 127 characters, you can continue entering commands and separate them with the Ctrl-T sequence. After you have entered all the commands, press Enter and each command is executed in turn.

### Erasing the entire command history

To start afresh and erase all the existing commands stored in the buffer, just press Alt-F7.

## Using DOSKEY Macros

Although too few people take advantage of it, DOSKEY has a useful macro facility that enables you to assign one or more commands to a single command. You can assign the following FORMAT command, for example, to the new command QFA:

```
format a: /v:technojock /q
```

When you enter QFA (for Quick Format A), DOSKEY substitutes the full format expression. This is just an example of how you can reduce potentially complex commands to one easy-to-use command. You define the shortcut command name and the command set that DOSKEY executes by using the following syntax:

```
doskey macroname=[text]
```

where *macroname* is any name you want to assign to the macro and *text* is a string of one or more commands. When the user enters *macroname*, DOSKEY executes *text*.

You create the preceding FORMAT example by using the following command:

```
doskey qfa=format a: /v:technojock /q
```

You can use any name for the macro, but it cannot include spaces or any of the redirection characters (<, >, and |, for example).

The method of removing or undefining a macro is similar to the syntax for clearing an environment entry. To remove a macro, execute DOSKEY and specify the macroname followed by the equal-sign character (=). To remove the QFA macro, for example, enter the following command:

```
doskey qfa=
```

Every time you reboot the computer, you must reinstall your macro commands. You should consider creating a batch file of your favorite macros and then call that batch file from AUTOEXEC.BAT after DOSKEY has been installed. A later section of this chapter, called "Building a macro builder," describes how you can build a batch file from the active macros.

NOTE

DOSKEY is capable only of substituting an entire command with a macro. You cannot use DOSKEY to substitute parts of a command. If you frequently copy or edit WIN.INI, for example, you might assign the single character W to represent WIN.INI by using the following statement:

```
doskey w=win.ini
```

Then, if you enter one of the following commands, you want DOSKEY to replace the *W* with WIN.INI:

```
edit w
copy w \winbak
```

Nice idea, but it doesn't work that way! You can replace only an entire command with a macro. In these examples, you end up editing or copying a file named W.

## Accessing command-line switches

Optional commands or switches can be passed to batch files by using the `%1` through `%9` identifiers. A similar approach is used with DOSKEY, except that the nine identifiers are `$1` through `$9` (a $ character is used in place of the % character).

The following macro creates a VC command (for Verify Copy) that adds a /V switch to the COPY command:

```
doskey vc=copy $1 $2 /v
```

If you were to execute the command `vc *.* a:`, DOSKEY would execute the command `copy *.* a: /v`.

In addition to the nine passed parameters, DOSKEY provides another identifier used to access the entire command line — namely, `$*`. When you specify `$*` in a command, DOSKEY substitutes all the switches and parameters passed on the command line, including spaces. This capability is particularly useful when user input can exceed nine parameters or when one or more parameters can include spaces, such as the prompt used in the CHOICE command.

The following example creates a macro called DF (for *def*rag), which executes the DEFRAG command and all the user-specified options and adds the /SKIPHIGH switch to force DEFRAG to load into conventional memory:

```
doskey df=defrag $* /skiphigh
```

I wish that batch files supported `%*`. Maybe next time.

## Piping and redirection with DOSKEY

For more information about redirection and piping, refer to Chapter 10.

To include any redirection or pipe symbols in a macro, you have to use special substitution characters that represent the real redirection characters. Imagine entering the following command to report the amount of free space in the current directory:

```
doskey dsize=dir | find "file(s)"
```

The intent of the command is to pipe the output from DIR to FIND, and FIND will echo any line that contains the string `"file(s)"` (the line that summarizes the total amount of space used by the files in the directory). When you enter DSIZE, a summary of the directory's files is displayed.

If you try to enter this command, however, you don't get the expected results. DOS intercepts the pipe character (`|`) and immediately pipes the DOSKEY command to FIND. As a result, DOSKEY only stores the command DIR with DSIZE.

The solution is to use the string `$b`. DOSKEY interprets it as the pipe character, but DOS ignores it when you first enter the macro-definition command. The correct way to enter the command, therefore, is as follows:

```
doskey dsize=dir $b find "file(s)"
```

There are similar substitution characters for each of the redirection commands (see Table 9-2). Note that the characters can be used in upper- or lowercase.

NOTE

If you want to use the dollar character in a command, enter two dollar characters ($$). This command stops DOSKEY from interpreting the character following the $ as a substitution character.

**Table 9-2 Substitutions for the Redirection and Pipe Characters**

| *String* | *Meaning* |
|---|---|
| `$b` | The equivalent of \|, which pipes the output from one command as input to another command |
| `$g` | The equivalent of >, which redirects output to another command, device, or file |
| `$gg` | The equivalent of >>, which appends output to the end of a file |
| `$l` | The equivalent of <, which redirects the contents of a file as input to a command |

## Entering multiple commands

When DOSKEY is installed, it enables you to enter multiple commands on the command line by separating every command sequence with the Ctrl-T character. To use multiple commands in a DOSKEY macro, a similar approach is used. Use the substitution characters $t, however, rather than press Ctrl-T. The following command, for example, creates the macro MCD (for Make and Change Directory):

```
doskey mcd=md $1$tcd $1
```

When you enter MCD followed by a path, the MD command is executed to make the directory, and then the CD command is executed to change to the directory.

## Building a macro builder

To review the current DOSKEY macro settings, use the /MACROS switch as follows:

```
doskey /macros
```

DOSKEY responds by listing all the installed macros, as shown in this example:

```
DSIZE=dir $b find "file(s)"
?=help $*
MCD=md $1$tcd $1
QFA=format a: /v:technojock /q
VC=copy $1 $2 /v
DF=defrag $* /skiphigh
```

Every time you reboot the computer or execute DOSKEY with the /REINSTALL switch, the active macros are lost. You can use the /MACROS switch as a tool to help save the existing macro settings so that they can be easily reinstalled.

By piping the output to a batch file, you have the framework of a DOSKEY macro builder. Executing the following command, for example, creates the batch file MACROS.BAT:

```
doskey /macros > macros.bat
```

The batch file has to be modified so that it can load the DOSKEY macros. Every line must be prefixed by the text `doskey` (with a space at the end). One way to make the changes is to edit the file MACROS.BAT manually; because computers are designed to automate repetitive tasks, however, why not let the computer do it?

The free program PREFIX (written by yours truly) is included on Bob's Shareware Collection disks. PREFIX reads lines from one ASCII file and writes them to another but prefixes every line with a specified command. The full syntax for the command is explained in Chapter 10. Its use in this context, however, is straightforward. To create a new batch file, SETMACRO.BAT, by taking the input from MACROS.BAT and prefixing every line with doskey (with a space at the end), enter the following command:

```
prefix macros.bat setmacro.bat doskey
```

PREFIX then creates the batch file SETMACRO.BAT with the following contents:

```
doskey DSIZE=dir $b find "file(s)"
doskey ?=help $*
doskey MCD=md $1$tcd $1
doskey QFA=format a: /v:technojock /q
doskey VC=copy $1 $2 /v
doskey DF=defrag $* /skiphigh
```

When you execute SETMACRO, the original macro commands are installed. Magic.

## Comparing DOSKEY Macros to Batch Files

The macro facility is similar in many ways to batch files. Both techniques allow a set of commands to be stored and then executed by invoking a shorter, simpler command, and both techniques allow the passing of command-line parameters. There are some important differences, however, and understanding them can help you to decide which option is best: macros or batch files.

This list shows the most significant differences between batch files and macros:

- Unlike batch files, macros are stored in memory, not in a file, so a macro command can be executed from any directory, regardless of the path.
- Even the smallest batch file consumes at least one cluster of disk space (usually 4,096 bytes), but macros use no disk space.

- Because macros are loaded in memory, they operate much faster than do batch files.
- A macro cannot call another macro command, but a batch file can call another batch file.
- A macro can call a batch file, but a batch file cannot invoke a macro command. Only commands entered directly on the command line are accessed by DOSKEY.
- You cannot suppress the echoing of macro commands, so the user can always observe macro commands as they execute (there is no echo off facility).
- DOSKEY does not support the IF or GOTO statements, so no conditional branching can be performed.
- Macros cannot access environment variables. You cannot access the path, for example, by using the `%path%` expression.
- Both techniques can access nine passed parameters, but macros can access the entire command line by using `$*` — batch files have no equivalent.
- Macro command names can be the same as internal DOS commands in addition to program files (with an EXE or a COM extension). The macro command is always executed in preference to the DOS command or external program of the same name. Batch files are ignored, however, if they have the same name as an internal DOS command or another program in the current directory.
- An entire macro is limited to 127 characters, but a batch file can be any length.

## My Favorite DOSKEY Macro Tricks

DOSKEY macros really are useful and can improve your productivity and safety if you take the time to build some valuable macros. This section discusses some of my favorites.

### Question-mark help

OK, this isn't rocket science, but I like to use a question-mark character rather than the HELP command. The following macro shows how:

```
doskey ?=help $*
```

Whenever you enter ?, the DOS HELP command is launched.

## Replacing, enhancing, and disabling DOS commands

A DOSKEY macro command takes priority over all other commands, whether they are internal DOS commands or external programs. In other words, the macro command is always executed. This feature enables you to disable selected commands or force additional command switches to be processed.

The following macro command, for example, disables the FORMAT command:

```
doskey format=The format command is disabled.
```

If the user enters the FORMAT command, DOSKEY executes the command `The format command is disabled`. The command is echoed to the screen, and DOS naturally responds with the infamous `bad command or file name` message (see Figure 9-2).

The following command adds the /P switch to Del to make sure that the user is prompted for confirmation before deleting a file:

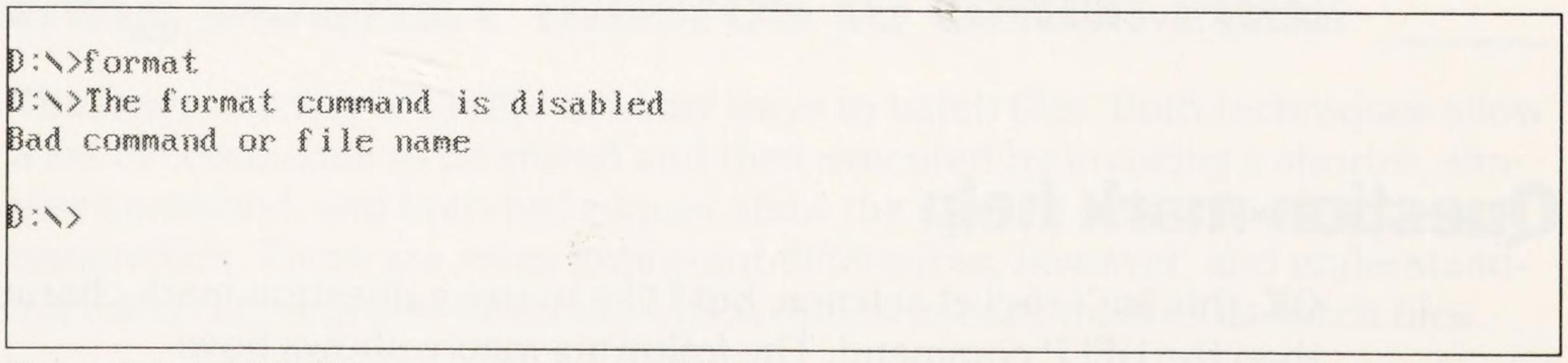
```
doskey del=del $1 /p
```

By using a similar technique, the following macro creates a command that substitutes an XCOPY command in place of COPY:

```
doskey copy=xcopy $*
```

Most of the macros discussed in this section are tools that enable you to configure PCs for other (less experienced) users to help them avoid "shooting themselves in the foot." These DOSKEY macros might be loaded from AUTOEXEC.BAT. To prevent simple tampering, you might also disable the DOSKEY command by entering the following line:

```
doskey doskey=The command is not available.
```



```
D:\>format
D:\>The format command is disabled
Bad command or file name

D:\>
```

**Figure 9-2:** A DOSKEY macro can disable DOS commands of the same name, like FORMAT.

None of these techniques is foolproof or, more accurately, Jedi-proof. A user can simply erase all the defined macros by pressing Alt-F10. Another trick is to enter a DOS command preceded by a space. Enter the command `format` (preceded by a space), for example, rather than `format` (with no space). DOSKEY ignores the command because it does not exactly match a macro name, and the command is passed directly to DOS. DOS ignores the preceding space and the FORMAT command (proper) is executed.

## Intercepting drive changes

My main desktop PC has a 1.2MB drive in drive A and a 1.44MB drive in drive B. My brain is programmed to enter `b:` to swap to the 3½-inch drive. When I use my notebook PC, I am forever entering `b:` when I mean `a:`. My solution is to assign the macro command `a:` to the macro name `b:`, as follows:

```
doskey b:=a:
```

Whenever I enter `b:`, DOSKEY intercepts it and enters `a:` instead. To try to train myself to end the habit of entering the wrong drive letter, I used to use the following macro:

```
doskey b:=echo ^G $ta:
```

You should enter the ^G string by pressing Ctrl-G, not by entering the two separate characters.

This macro beeped at me and then changed to drive A, but it irritated me too much, so I changed it back.

You can use a similar approach for a network drive. The following macro, for example, checks whether drive F exists; if it does not exist, a log-on command is issued before changing to drive F. Otherwise, the F: command simply is executed:

```
doskey f:=if not exist f: net logon$tf:
```

The concept of using IF and EXIST to check whether a directory exists is discussed more in Chapter 10.

## Supporting multiple filenames

Some of the most popular commands support only one filemask (Del). By using a FOR loop in a macro, you can expand these types of commands to accept multiple filemasks. Consider the following macro definition:

```
doskey delall=for %n in ($*) do del %n
```

When you pass multiple filenames or masks to the command DELALL, the DEL command is called once for every separate parameter. To delete all files with the extensions of DEL, BAK, TMP, and $$$, for example, you enter the following command:

```
delall *.del *.bak *.tmp *.$$$
```

The following macro uses the same technique to enhance the COPY command to support multiple filemasks:

```
doskey copyto=for %f in ($2 $3 $4 $5 $6 $7 $8 $9) do copy %f $1
```

The COPYTO command accepts the target directory as the first argument, and the remaining arguments are the source filemasks. To copy a disparate group of files to drive A, for example, you enter a command similar to the following:

```
copyto a: *.doc *.tif *.txt \wks*.wk3
```

## Extending the search path

I hate to have a huge path. It uses up the environment and slows down the system while DOS searches every path's directory for a command. If you have a directory in the path solely to make it easy to execute a specific command from any other directory, consider creating a DOSKEY macro entry instead of clogging the PATH. To launch the CMFILER program, for example, you might create the following macro:

```
doskey cmfiler=c:\utils\cm\cmfiler
```

No matter which drive and directory is the default, when you enter the command CMFILER, the program is loaded from the directory C:\UTILS\CM.

## And some quickies to finish

Here are three one-character macros I use for common tasks:

I like to enter the backslash character (\) to change the directory to the root directory. The following macro makes this possible:

```
doskey \=cd\
```

I am forever editing my CONFIG.SYS and AUTOEXEC.BAT files. The following two macros launch EDIT and load a start-up file after the press of a button:

```
doskey c=edit c:\config.sys
doskey b=edit c:\autoexec.bat
```

## Going Beyond DOSKEY with ANARKEY

If you like DOSKEY, you will love ANARKEY, from Moderne Software. ANARKEY is included on Bob's Shareware Collection disks and is documented in Part V of this book.

ANARKEY, a direct replacement for DOSKEY, offers all the standard DOSKEY functions and then some. The following section summarizes ANARKEY's main features and is organized by its four primary functions: command-line editing, environment editing, history lists, and macros.

### Command-line editing

Like DOSKEY, ANARKEY provides command-line editing that enables you to move the cursor along the command line and change and delete characters as you go. ANARKEY goes beyond simple editing, however, by offering file and directory completion.

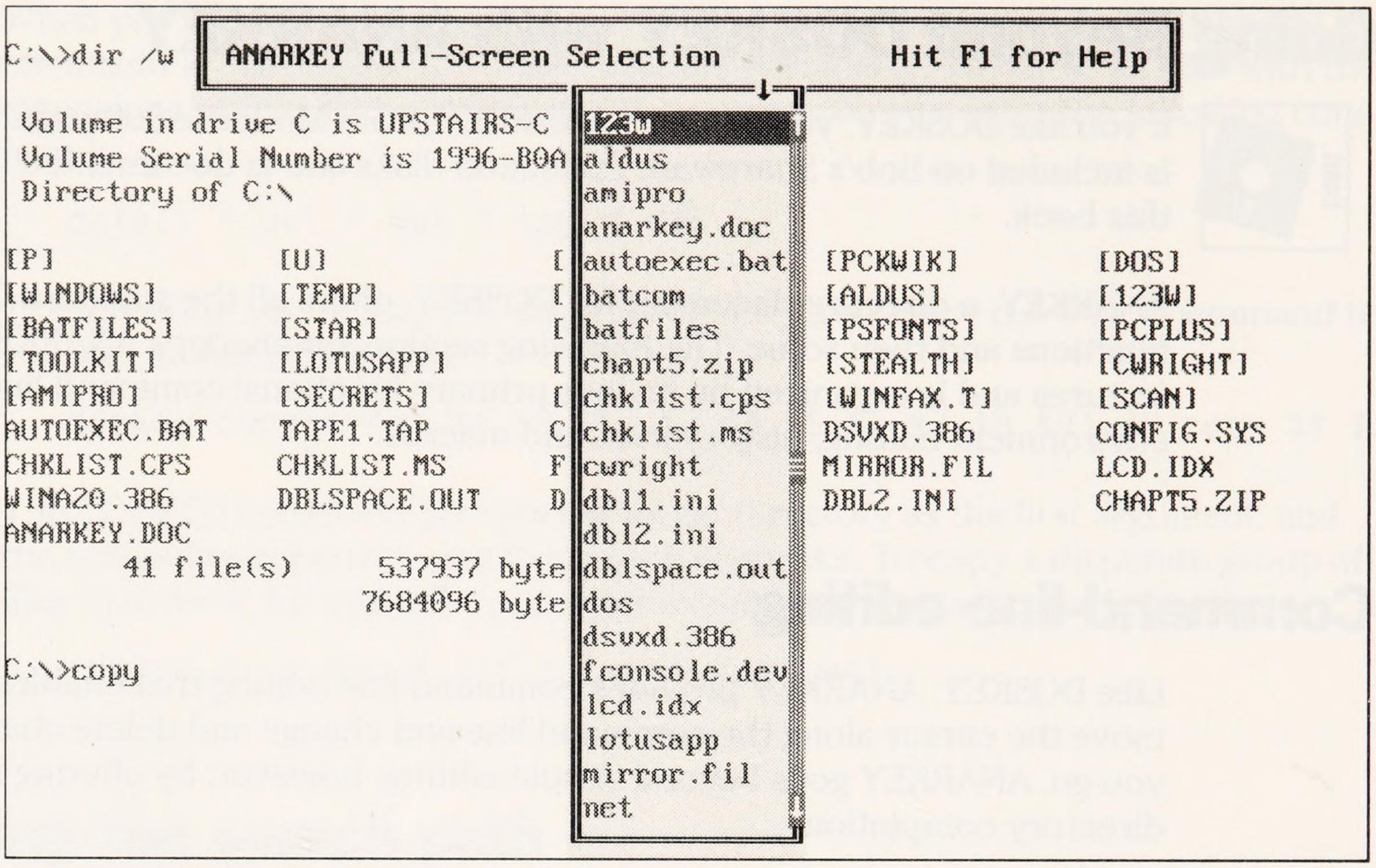
If you type the command COPY and then press F8 (or Ctrl-F), for example, ANARKEY searches the current directory and puts the name of the first file directly on the command line. This feature is known as *filename completion*. You can press F8 additional times to toggle through all the filenames. Better still, if you enter a command with a partial filename and then press F8, ANARKEY reports only files that match the partial name. Type the following command, for example, and press F8:

```
edit c:\co
```

ANARKEY completes the command as `edit c:\config.sys`.

Similarly, you can use ANARKEY to complete a directory name. Suppose that you enter the command `cd\`. If you press F7 (or Ctrl-D), ANARKEY inserts the first directory in the root directory. You can scroll through the directories quickly by pressing F7 additional times. As with filename completion, you can enter a partial directory name to focus ANARKEY's directory selection.

Better still, if you enter a partial command and then press Alt-F8 (or Alt-F), ANARKEY displays a pop-up selection Window (see Figure 9-3). Like all of ANARKEY's windows, the Filename Completion window offers full mouse support.



```
C:\>dir /w   ANARKEY Full-Screen Selection             Hit F1 for Help

 Volume in drive C is UPSTAIRS-C 123w
 Volume Serial Number is 1996-B0A aldus
 Directory of C:\                amipro
                                 anarkey.doc
[P]            [U]             [ autoexec.bat  [PCKWIK]      [DOS]
[WINDOWS]      [TEMP]          [ batcom        [ALDUS]       [123W]
[BATFILES]     [STAR]          [ batfiles      [PSFONTS]     [PCPLUS]
[TOOLKIT]      [LOTUSAPP]      [ chapt5.zip    [STEALTH]     [CWRIGHT]
[AMIPRO]       [SECRETS]       [ chklist.cps   [WINFAX.T]    [SCAN]
AUTOEXEC.BAT   TAPE1.TAP       C chklist.ms    DSVXD.386     CONFIG.SYS
CHKLIST.CPS    CHKLIST.MS      F cwright       MIRROR.FIL    LCD.IDX
WINA20.386     DBLSPACE.OUT    D dbl1.ini      DBL2.INI      CHAPT5.ZIP
ANARKEY.DOC                      dbl2.ini
      41 file(s)      537937 byte dblspace.out
                     7684096 byte dos
                                 dsvxd.386
C:\>copy                         fconsole.dev
                                 lcd.idx
                                 lotusapp
                                 mirror.fil
                                 net
```

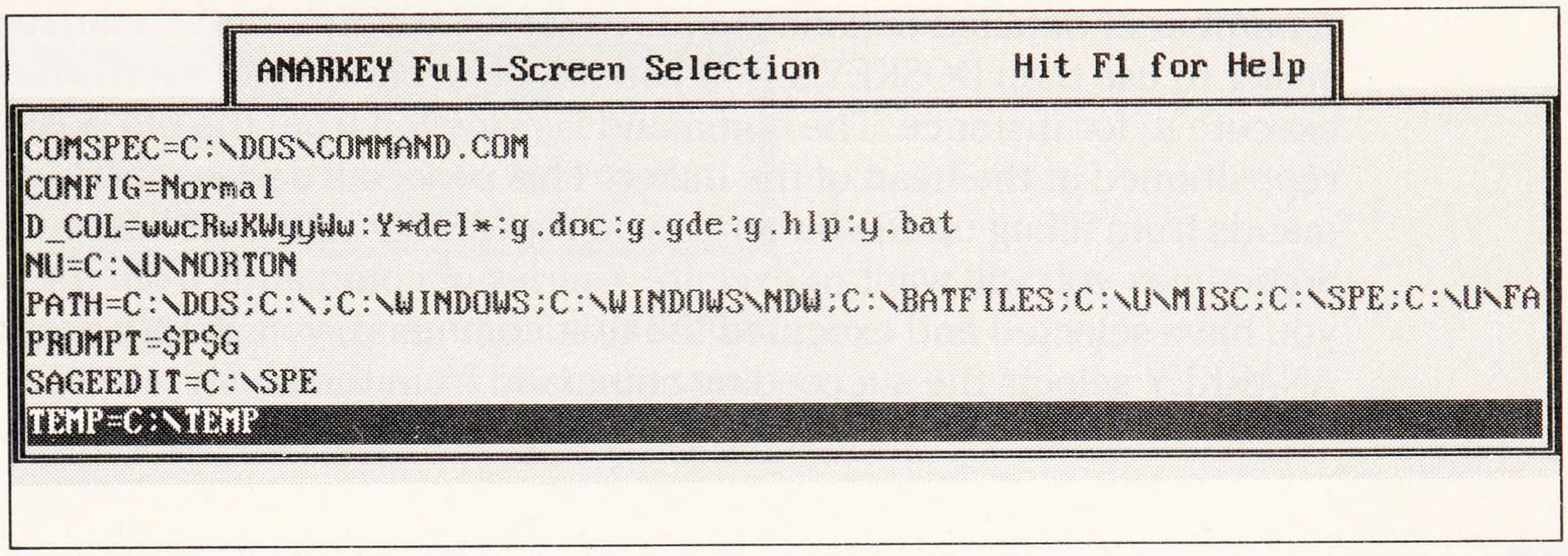
**Figure 9-3**: ANARKEY displays a pop-up window that lists all matching filenames and directories when you press Alt-F8.

## Environment editing

A useful extension to the command-line editing feature is the ability to edit environment strings. To edit an environment variable, enter the command `set` and then press F9 (or Ctrl-E). ANARKEY places on the command line the first variable in the environment. You can cycle through all the environment variables by pressing F9 additional times.

If you want to edit a specific variable and you know some or all of the environment name, just type the `set` command followed by the environment name (or the first part of it) and press F9. ANARKEY searches the environment and displays the first matching variable. To change the PATH command, for example, enter the command `set path` and press F9. ANARKEY moves the complete path statement to the command line, where you can edit it and press Enter to change it.

ANARKEY also has a pop-up environment window (see Figure 9-4) that is displayed when you press Alt-F9. The window enables you to scroll through the entire environment and select the statement you want to edit.



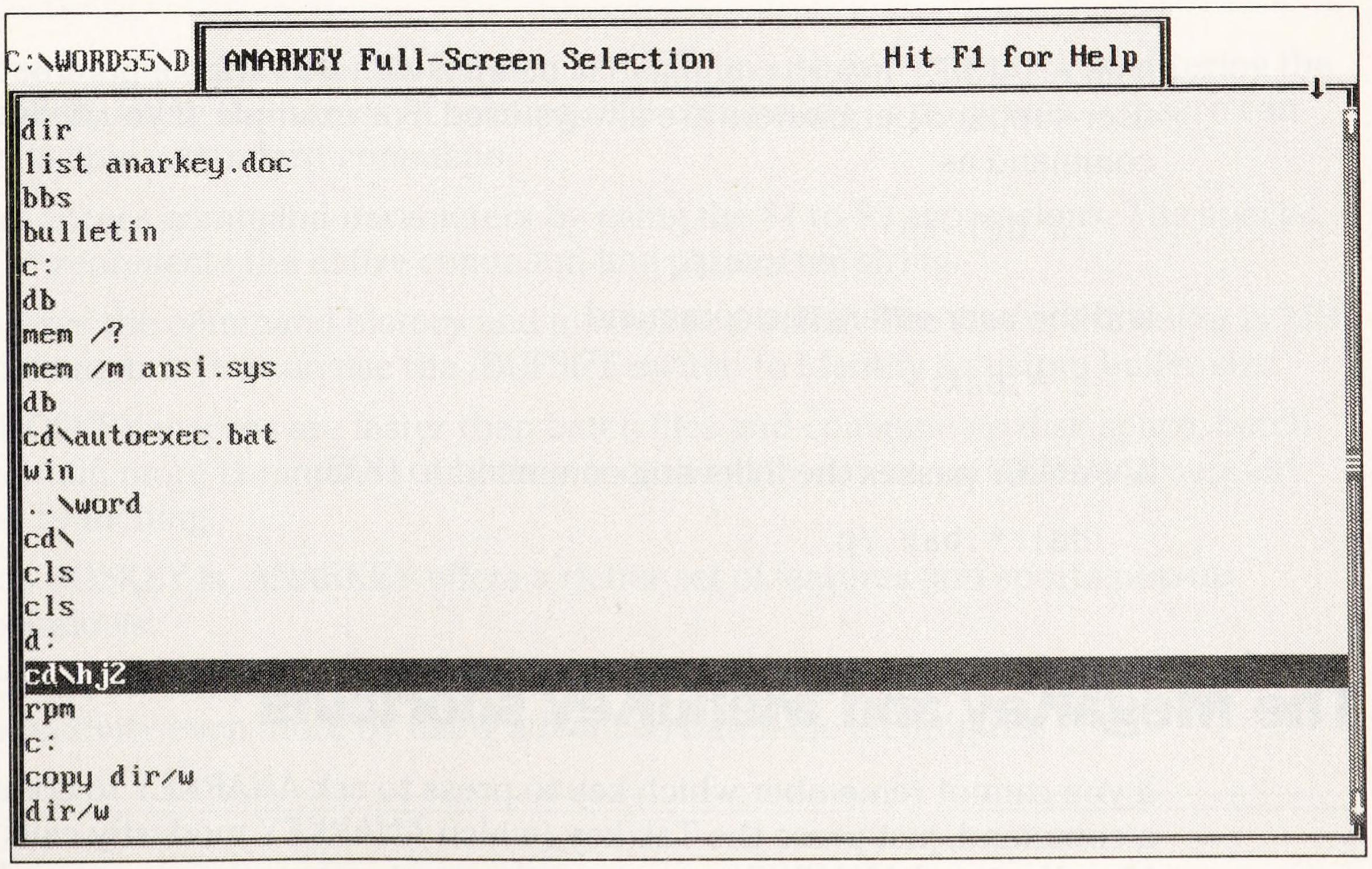
```
COMSPEC=C:\DOS\COMMAND.COM
CONFIG=Normal
D_COL=wwcRwKWyyWw:Y*del*:g.doc:g.gde:g.hlp:y.bat
NU=C:\U\NORTON
PATH=C:\DOS;C:\;C:\WINDOWS;C:\WINDOWS\NDW;C:\BATFILES;C:\U\MISC;C:\SPE;C:\U\FA
PROMPT=$P$G
SAGEEDIT=C:\SPE
TEMP=C:\TEMP
```

**Figure 9-4:** ANARKEY displays a pop-up window that lists all the environment variables when you press Alt-F9.

## History lists

Just like DOSKEY, ANARKEY maintains a history list of all the commands you have entered since ANARKEY was loaded. By pressing the up- and down-arrow keys, you can cycle through these old commands.

As you might expect, ANARKEY implements a number of features that make it more functional and useful than DOSKEY. The most visible feature is that you can display a pop-up window of the old commands by pressing Alt-F5 (or Alt-K). Figure 9-5 shows an example of a pop-up history list. You can use the arrow keys or the mouse to scroll through the commands and then press Enter to insert the command on the command line.



```
dir
list anarkey.doc
bbs
bulletin
c:
db
mem /?
mem /m ansi.sys
db
cd\autoexec.bat
win
..\word
cd\
cls
cls
d:
cd\hj2
rpm
c:
copy dir/w
dir/w
```

**Figure 9-5:** Press Alt-F5 to display a history list of all the old commands.

A number of detailed enhancements make ANARKEY's buffer management easier to use than DOSKEY's — if you select a command from the buffer and execute it, for instance. The command is removed from the old position and repositioned at the head of the buffer. This procedure prevents duplicate commands from filling up the buffer as you repeatedly select and execute them. Sometimes you will want to execute a series of commands from the buffer. After you have selected and executed the first command, you simply press PgDn and ANARKEY selects the succeeding command from the buffer.

## Macros

The features provided in ANARKEY's macro facility (referred to as AKA in the program documentation) are similar to those in DOSKEY's macro facility, but they use a different syntax. To install ANARKEY with macro support, you must execute ANARKEY with the -A*nnn* switch, where *nnn* is the amount of memory, in bytes, to allocate to macros.

To create an ANARKEY macro, use the following syntax:

:*macroname* [*text*]

You can view a list of the installed macros by pressing Alt-F4, and you can erase the macros by pressing Alt-X.

DOSKEY and ANARKEY macros treat passed parameters a little differently. ANARKEY does not provide a feature to access the entire command line (like DOSKEY's `$*` switch); access to the nine passed parameters is by way of the symbols from `%1` to `%9`, as in batch files, rather than from `$1` to `$9`.

If an ANARKEY macro command is defined with no variable parameters, the user-supplied parameters are always used. For example, if you define the D command as

```
:d del /p
```

and the user enters the command

```
d *.bak
```

ANARKEY passes the following command to DOS:

```
del *.bak /p
```

## The MegaKey and MenuKey shortcuts

If you cannot remember which key to press to ask ANARKEY to complete a command, just press the Tab key (which ANARKEY modestly calls the MegaKey) and ANARKEY guesses at the most appropriate command-

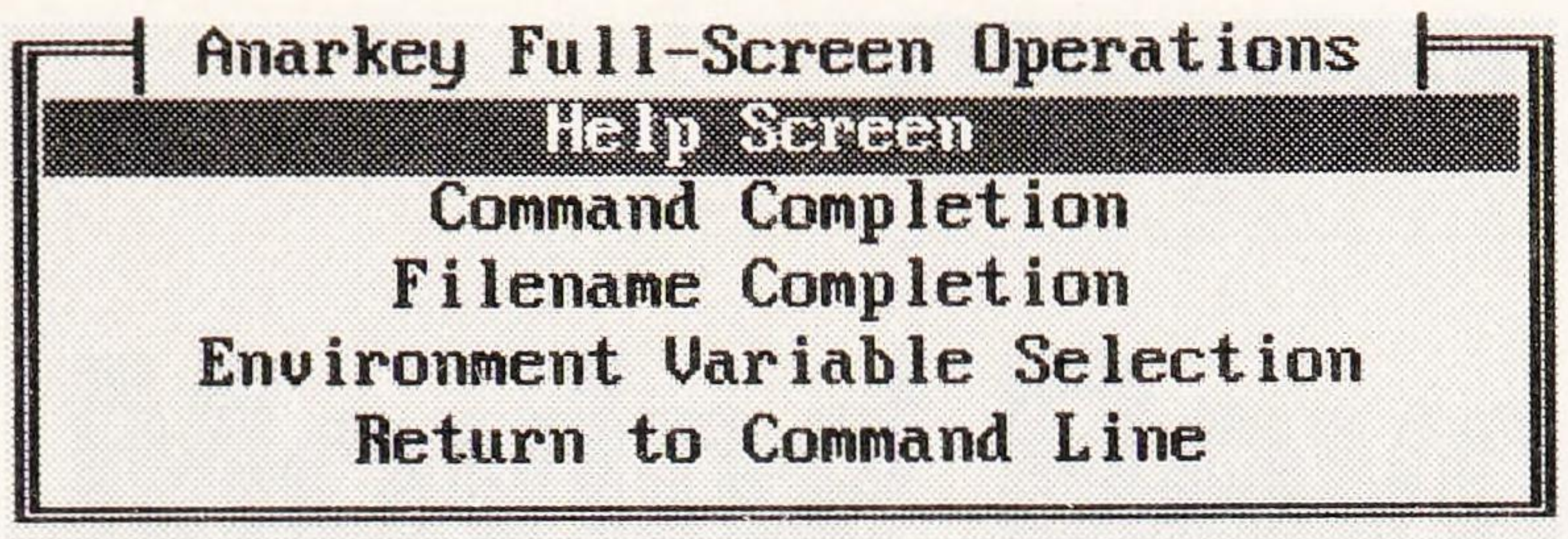


**Figure 9-6:** When you press Shift-Tab, ANARKEY displays a menu that provides quick access to each pop-up window.

completion technique to use (filename, directory name, or a command from the history list, for example).

Similarly, if you cannot remember which key pops up which window, just press Shift-Tab and ANARKEY displays a menu of all the windows (see Figure 9-6).

## *Summary*

This chapter described the many uses of DOSKEY and showed how it can increase your productivity by automating tasks. The chapter discussed the following points:

- In addition to providing basic line editing, DOSKEY permits old commands to be recalled by using the up- and down-arrow keys.
- Old commands also can be displayed and reclaimed from the history list by using the lesser known F7 and F9 functions.
- Multiple commands can be entered on one line, if every block is separated by the Ctrl-T character.
- One or more commands can be stored as a DOSKEY macro and executed by entering the macro name. By using the names of existing DOS commands or programs, a macro can replace or disable a standard command.
- Macros can access command parameters by using the $1 to $9 expressions. The special sequence $* represents the entire command-line parameter string.
- DOSKEY stores the command history and macros in a buffer. The size of the buffer is 512 bytes by default, but you can use the /BUFSIZE switch to identify a custom buffer size.
- Although DOSKEY macros are faster than batch files and consume no disk space, batch files can contain more commands, and they access environment variables and support conditional branching.
- As useful as DOSKEY is, ANARKEY offers a richer set of features and sports pop-up selection windows.

In many respects, batch files pick up where DOSKEY leaves off. Chapter 10 explains how to increase productivity even more by using advanced batch-file techniques.

# Chapter 10

# Moving Beyond Basic Batch Files

**In This Chapter**

- Applying the tools of the trade
- Using power techniques with redirection and pipes
- Using environment innovations
- Determining dates and times with batch files
- A batch file potpourri
- Using batch files with Windows 3.1
- Enhancing batch files

## The Tools of the Trade

With more than 100 commands, many with five or more switches, it's no wonder that few people take advantage of the wealth of features in DOS. The key to harnessing the power of DOS is within batch files.

AUTOEXEC.BAT, the mother of all batch files, is a great illustration of how batch files simplify DOS. Just imagine having to enter all those start-up commands every time you boot your computer. Unthinkable. It is a shame that many users don't take the time to automate tasks by creating batch files. A DOS Jedi knows better.

No one would describe the DOS batch facility as a programming language; C or Pascal it isn't. The batch file tools provided with DOS are crude, in fact. But with a little imagination and trickery, you can use these simple tools to create powerful and useful utilities that will make you more productive.

## A Crash Course on Batch Files

Batch files are ASCII files that contain one or more commands. Every batch file name has an extension of BAT (short for batch). You execute all the commands listed in the batch file by simply entering the name of the batch file at the command prompt. Think of a batch file as a script or list of commands that can be executed in one easy step.

The following simple batch file, MCD1.BAT, creates the directory \DEMO with the MD command and then uses CD to change the default directory to \DEMO:

```
md \demo

cd \demo
```

You could create this batch file by running EDIT and entering the two lines of commands. To run this batch file, just enter this command:

```
c:\> mcd1
```

DOS, in turn, executes the MD and CD statements.

Batch files can do more than process standard commands, and several DOS commands are designed specifically for use in batch files. Consider the following batch file, MCD2.BAT:

```
@echo off

rem MCD2 by R D Ainsbury

rem Makes a \DEMO directory and changes to it

md \demo

cd \demo
```

The `@echo off` line turns off command echoing to reduce screen clutter, and the `rem` command is used to add comments to the file. The example batch file is still very limited because it creates only the specific directory \DEMO. You can, however, easily improve it to create any valid directory. Like many DOS commands, batch files can be executed with optional switches; within the batch file, these switches are represented by the variables %1 through %9. The enhanced version of the batch file, MCD3.BAT, uses the %1 variable to represent the first item specified after the batch filename on the command line:

```
@echo off

rem MCD3 by R D Ainsbury

rem Makes a directory and changes to it

md %1

cd %1
```

To create and change to any directory, just enter the command MCD3 and then the directory name. The directory automatically is created and made the default. To create the directory \STUFF, for example, you just enter the following command:

```
c:\> mcd3 \stuff
```

You can improve this batch file with error checking and input validation. If you use an asterisk in the directory name, for example, DOS cannot create the directory. Look at the example for the IF command in Part IV, "The Complete DOS 6 Command Reference," for an example of how to improve error checking by using other batch commands, such as IF and GOTO.

# A batch file command summary

Table 10-1 contains an alphabetical list of the DOS 6 batch file commands and a brief description of each command.

In addition to the standard batch file commands, a few other DOS keywords and commands are often utilized in batch files (see Table 10-2).

### Table 10-1 DOS 6 Batch File Command Summary

| *Command* | *Description* |
|---|---|
| @ | Prevents the remainder of the command line from being echoed |
| :LABEL | Identifies in a batch file a location where processing can branch to by using the GOTO command |
| CALL | Launches a second batch file; when the second batch file is completed, control returns to the calling batch file |
| CHOICE | Displays a prompt and waits for the user to press one key from a set of keys |
| ECHO | Displays a string or toggles the command-echoing status |
| FOR | Executes a command or statement multiple times and applies a new string or filename with every iteration |
| GOTO | Branches command processing to a specific location (label) in the batch file |
| IF | Provides a conditional branch in a batch file execution; can be used to compare two strings, test the ERRORLEVEL returned by a program, or test whether a file exists |
| PAUSE | Temporarily halts batch file execution and waits for any key to be pressed |
| REM | REM, short for remark, is designed for adding comments to batch files (and to CONFIG.SYS); any text on a line that starts with REM is ignored |
| SHIFT | Allows access from a batch file to more than nine passed parameters by dropping the first passed parameter and shifting all other parameters to the left one position |

### Table 10-2 Other Batch File Tools

| *Item* | *Description* |
|---|---|
| FIND | Searches for the presence or absence of specific text. |
| SORT | Sorts ASCII data and directs the sorted output to a device or file. |
| %0..%9 | Variables entered on the command line when the batch file was executed. Initially, %0 contains the name of the batch file, %1 contains the first parameter, %2 contains the second parameter, and so on. |
| %ENVVAR% | A variable whose value is equal to the contents of the environment variable with the name *ENVVAR*. For example, in a batch file, %PATH% has the value of the full path, and %COMSPEC% contains the COMMAND.COM path and filename. |
| ERRORLEVEL | A value in the range from 0 to 255 that a program returns to indicate success or failure. You should use IF to test the ERRORLEVELs in reverse order. |

## It's about time we had a choice

I won't dwell on the basic use of the batch commands — you can refer to the command reference in Part IV for a full definition and description of each command. One new batch command, however, deserves some attention.

We've been waiting ten years for a DOS batch file command that prompts for input, and — finally — we have it. The CHOICE batch command is an external command that is less than 2,000 bytes in size. Because half the file contains help text, I wonder why it took so long!

This list shows the primary features of CHOICE:

- CHOICE displays an optional message and waits for the user to press a key.
- By default, CHOICE waits until either Y or N is pressed, but you can optionally specify a set of valid keys.
- You can enable case sensitivity, but keys in either case are allowed by default.
- An ERRORLEVEL is set to indicate which key the user pressed. Usually, a CHOICE statement is followed immediately by one or more IF statements.
- You can optionally specify a time-out period in the range from 0 to 99 seconds and a default key. After the time-out period has elapsed, CHOICE terminates and sets the ERRORLEVEL corresponding to the default key. Contrary to most other commands, CHOICE sets an ERRORLEVEL of 0 to indicate an error (a syntax error in the command or the user pressed Ctrl-Break, for example).

As useful as CHOICE is, it lacks a number of features:

- Even though CHOICE supports most standard keys (including ;, [, and *), it does not support the spacebar, function keys, or key combinations such as Alt-X.
- There is no easy way to set CHOICE to accept any key.
- CHOICE supports only single-key input — there is no convenient way to enter a word or phrase.

The following example shows CHOICE used in a batch file to prompt the user to choose the type of pointing device:

```
@echo off
choice /c:mtn /t:n,10 Do you want to load the Mouse, Tablet or Neither
if errorlevel 3 goto end
if errorlevel 2 goto Tablet
mouse /y
goto end
:Tablet
tabk /c:345
:end
```

For clarity, every batch file listing in this chapter includes line numbers. These line numbers are not part of the batch file — they are used as reference points in the text. If you create these batch files, do not add line numbers.

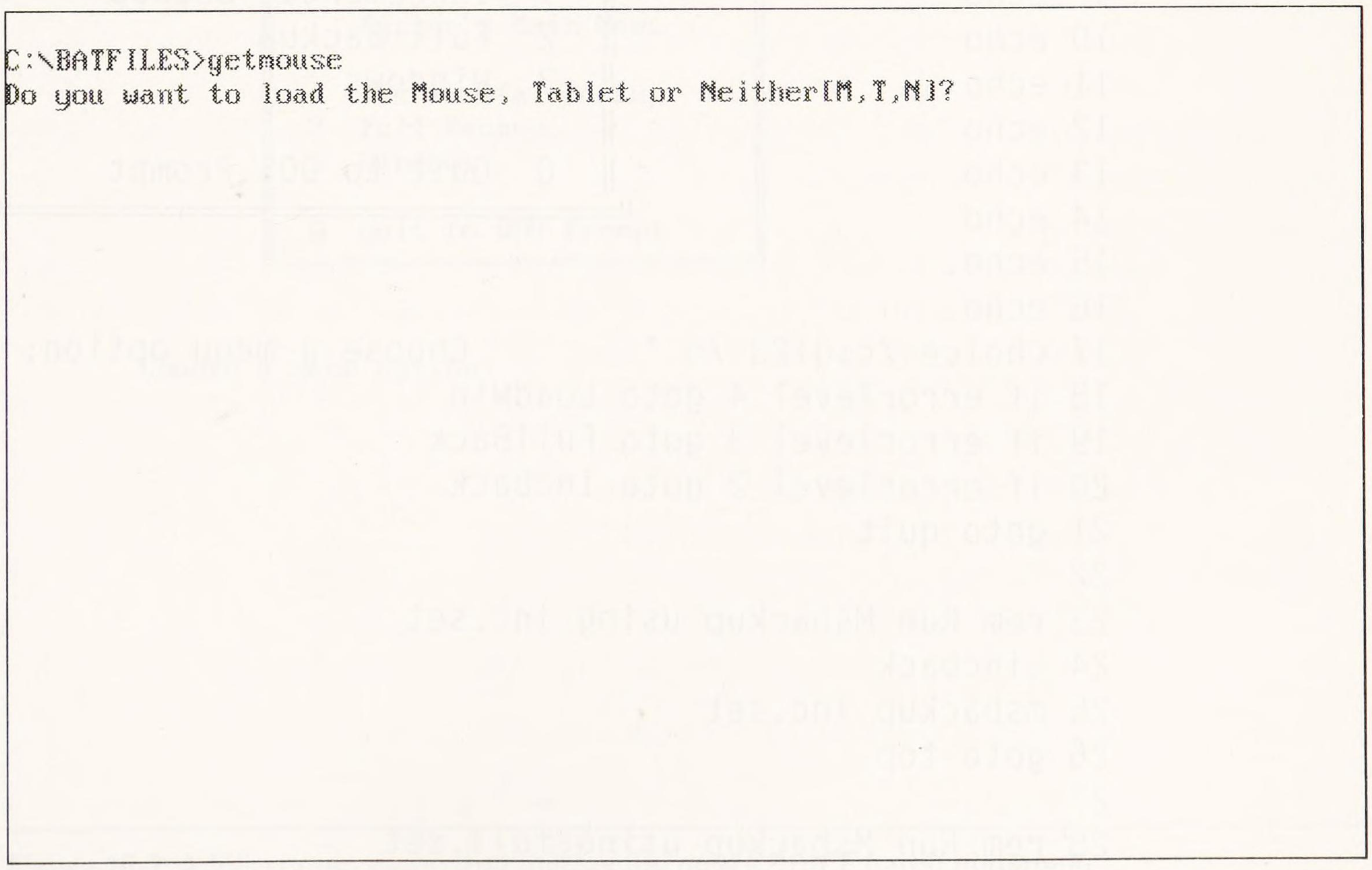
The /C:MTN switch on line 2 instructs CHOICE to accept any one of three characters: M, T, or N. Because the case-sensitivity switch, /S, was not specified, either upper- or lowercase letters are accepted.

The ERRORLEVELs returned from CHOICE are tested (in reverse order, of course) to determine which key the user pressed. The ERRORLEVELs correspond with the order in which the keys were specified by using the /C switch. In this case, a 3 is returned if the user pressed N, a 2 if the user pressed T, and a 1 if the user pressed M.

The /T:N,10 switch on line 2 instructs CHOICE to wait for ten seconds for input. If no key is pressed, CHOICE proceeds as though the user had pressed N.

Figure 10-1 shows the prompt generated when the batch file is executed. Notice that CHOICE automatically displays the valid keys in the format `[M,T,N]?`. You can suppress this prompt by using the /N switch.

The prompt that displays the valid keys is always located in the first character position following the message text. Add a space at the end of the CHOICE command line to separate the message text and the prompt.



```
C:\BATFILES>getmouse
Do you want to load the Mouse, Tablet or Neither[M,T,N]?
```

**Figure 10-1**: CHOICE displays some message text and a list of valid keys.

If you want to include a slash mark (/) in the message text, enclose the message in quotation marks. This action prevents CHOICE from trying to interpret as a switch the character following the /. If you use a / and don't include the quotation marks, you get one of the following messages (depending on which character follows the /):

```
Invalid switch on command line
```

or

```
Only one prompt string allowed
```

CAUTION

Do not use the pipe and filter characters (|, <, >, >>, and <<) in the CHOICE text, even if the text is enclosed in quotation marks.

CHOICE is ideal for building simple menu systems. The following example is a complete menu system contained in one batch file. Figure 10-2 shows the menu screen it produces.

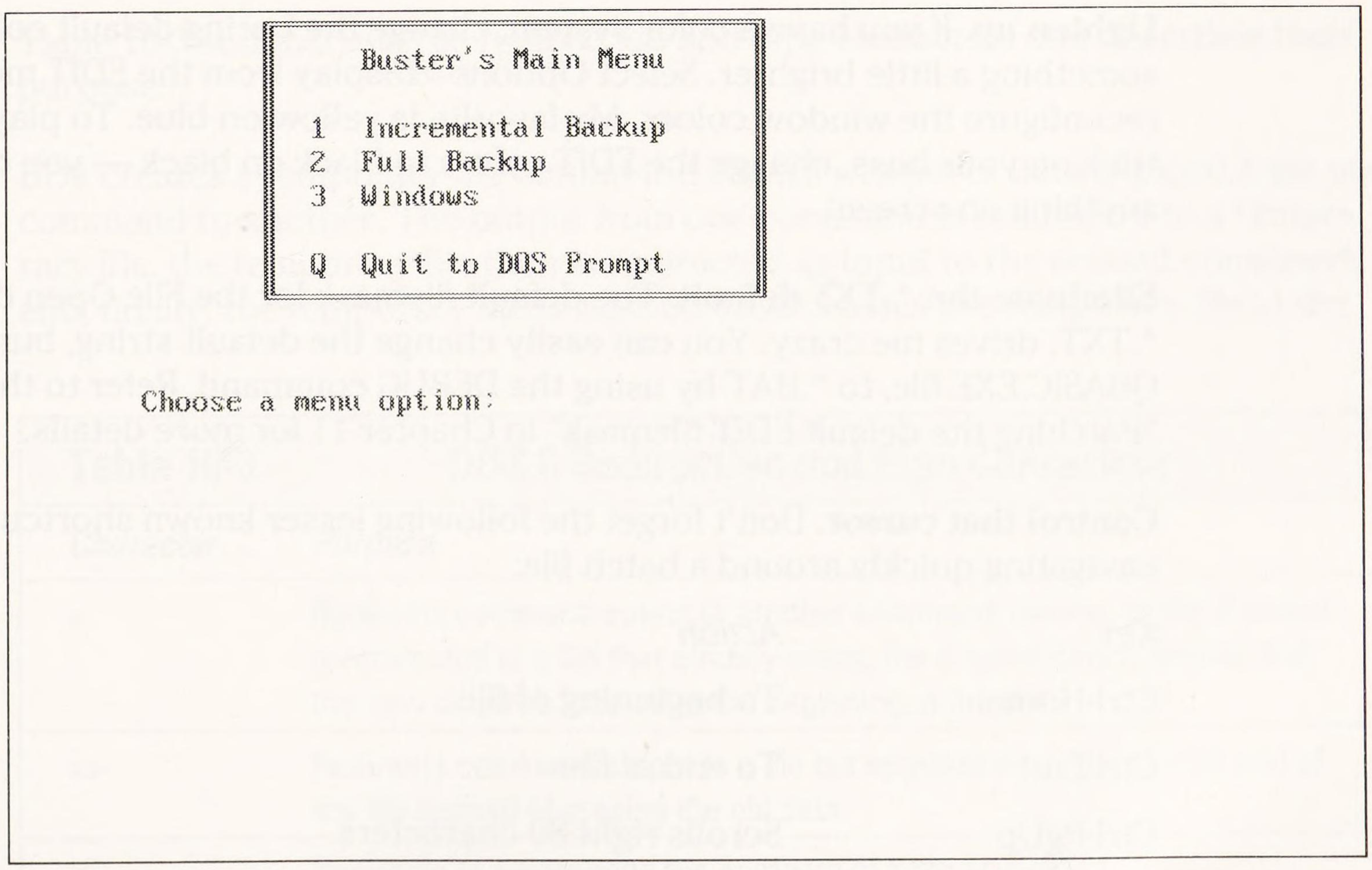
```
@echo off
:top
c:
cd\
cls
echo                ╓─────────────────────────────╖
echo                ║     Buster's Main Menu      ║
echo                ║                             ║
echo                ║  1  Incremental Backup      ║
echo                ║  2  Full Backup             ║
echo                ║  3  Windows                 ║
echo                ║                             ║
echo                ║  Q  Quit to DOS Prompt      ║
echo                ╙─────────────────────────────╜
echo.
echo.
choice /c:q123 /n "        Choose a menu option: "
if errorlevel 4 goto LoadWin
if errorlevel 3 goto FullBack
if errorlevel 2 goto IncBack
goto quit

rem Run Msbackup using inc.set
:incback
msbackup inc.set
goto top

rem Run Msbackup using full.set
```

```
:fullback
msbackup full.set
goto top

rem Change command prompt and run Windows
:loadwin
prompt Enter EXIT to return to Windows$_$p$g
win
prompt $p$g
goto top

:quit
```

## Editor tips

Most batch files are less than a page or two long. EDIT, the full-screen editor, provides a convenient way to create and edit batch files. Although EDIT supports the normal conventions for editing files, the tips in this section can increase your productivity:

**Typing special ASCII characters.** To enter a nonstandard ASCII character, such as the line-drawing characters used to draw boxes, you must enter the character's ASCII value. Appendix C in the *MS-DOS User's Guide* contains a



```
Buster's Main Menu

1  Incremental Backup
2  Full Backup
3  Windows

Q  Quit to DOS Prompt

Choose a menu option:
```

Figure 10-2: A full-screen menu that utilizes the new CHOICE batch command.

complete list of all 256 ASCII characters. After you have determined the three-digit ASCII code, follow these steps:

1. Make sure that Num Lock is enabled.
2. Hold down the Alt key.
3. Press the three digits on the numeric keypad.
4. Release the Alt key.

The special ASCII character will be inserted at the cursor position.

**Use QBASIC's Editor.** When you execute the EDIT command, it in turn executes QBASIC.EXE with the /EDITOR switch. The /EDITOR switch tells QBASIC to switch into edit-only mode so that many of the QBASIC-related menu options are hidden from view. Unfortunately, a couple of useful editing features also are suppressed. Try running QBASIC rather than EDIT to take advantage of the following two features:

- Use the **V**iew⇨**Sp**lit menu option to divide the screen into two separate areas. This feature enables you to view one area of the file while you are editing another area. You can press F6 to flip the cursor from window to window. Select **V**iew⇨**S**plit a second time to return to a single edit screen.
- Press F4 to view the DOS screen that was displayed when you launched QBASIC. This feature is very useful when you try to fix a bug in a batch file and you want to see the errors while you are editing the file.

**Lighten up.** If you have a color system, change the boring default colors to something a little brighter. Select **O**ptions⇨**D**isplay from the EDIT menu to reconfigure the window colors. My favorite is yellow on blue. To play a devious trick on your boss, change the EDIT colors to black on black — you can't see anything on-screen!

**Eliminate the *.TXT default.** The default filemask for the File Open dialog box, *.TXT, drives me crazy. You can easily change the default string, buried in the QBASIC.EXE file, to *.BAT by using the DEBUG command. Refer to the section "Patching the default EDIT filemask" in Chapter 11 for more details.

**Control that cursor.** Don't forget the following lesser known shortcut keys for navigating quickly around a batch file:

| *Key* | *Action* |
|---|---|
| Ctrl-Home | To beginning of file |
| Ctrl-End | To end of file |
| Ctrl-PgUp | Scrolls right 80 characters |
| Ctrl-PgDn | Scrolls left 80 characters |
| Ctrl-up arrow | Scrolls up a line |

| | |
|---|---|
| Ctrl-down arrow | Scrolls down a line |
| Ctrl-left arrow | To the left one word |
| Ctrl-right arrow | To the right one word |
| Ctrl-Enter | To beginning of next line (without a carriage return) |
| Ctrl-Q,E | To top of window |
| Ctrl-Q,X | To bottom of window |

DISK

As useful as EDIT is, there are many features it doesn't support. For example, there is no quick way to draw boxes, you can edit only one file at a time, you cannot easily import text from another file, and it does not support block column moves. If you are looking for an editor that has these additional features and much more, try Semware's excellent shareware editor, QEDIT. Refer to Part V of this book for a full description of QEDIT's features.

## Power Techniques That Use Redirection and Pipes

The DOS redirection and piping characters play an important role in advanced batch file programming. You can, for example, pass characters to commands, create temporary files, and search or suppress command output.

Table 10-3 summarizes the redirection and pipe characters and describes their purpose.

GEM

DOS creates a temporary file behind the scenes whenever data is piped from one command to another. The output from one command is redirected to a temporary file, the temporary file then is redirected as input to the second command, and, finally, the temporary file is deleted. DOS creates the temporary file in the

**Table 10-3 DOS 6 Redirection and Pipe Characters**

| *Character* | *Purpose* |
|---|---|
| > | Redirects command output to another command, device, or file; if output is redirected to a file that already exists, the original data is erased and the new output is located at the beginning of the file |
| >> | Redirects command output to a file but appends the output to the end of the file instead of erasing the old data |
| < | Redirects to a command the contents of a file as input |
| \| | Pipes the output from one command as input to another command |

directory specified with the TEMP environment variable. If no such variable exists, the file is created in the current directory.

The following section looks at a variety of real-life applications for the redirection and piping tools.

## Just say yes

Some DOS commands have a habit of prompting for confirmation before proceeding with a task. For example, when you enter the command

```
del *.*
```

DOS displays this message:

```
Are you sure (Y/N)?
```

Although this message provides a valid safety feature for command-line execution, it can be annoying when a batch file pauses for confirmation part of the way through a series of commands.

By piping the output from ECHO to a command, you can trick the command into believing that a key was pressed. The following batch file shows this technique used with the DEL command:

```
@echo off
echo y | del *.*
```

**CAUTION** If you create this batch file, make sure that you run it from a temporary directory. Don't forget that it deletes every file in the current directory and doesn't prompt for confirmation. Alternatively, you can change the `echo y` statement to `echo n`.

**DISK** The use of ECHO to pipe keystrokes to a command is useful but very limited. Bob's Shareware Collection disks include the champion of key stuffers — STACKEY, from CtrlAlt Associates. This small TSR enables you to pass almost any key, key combination, or string to an application.

The following EDITBAT.BAT batch file, for example, uses STACKEY to store some keystrokes and then start EDIT. As soon as EDIT starts, STACKEY feeds these keystrokes to the editor. In this example, the keystrokes create the beginning of a new batch file:

```
@echo off
stackey W36 ES "@echo off" CR "REM Written by R D Ainsbury" CR
if not "%1"=="" stackey @F "S" "%1.bat" CR
edit
```

Line 2 instructs STACKEY to wait 36 clock ticks (2 seconds) before passing the keystrokes to the application. The ES code represents the Esc key and is used to remove the initial dialog box that EDIT displays when no filename is specified. The code `CR` (for *c*arriage *r*eturn) represents the Enter key. Line 3 checks to see whether the batch file was passed a parameter; if so, the keys Alt-F,S are passed to the editor to invoke the File⇨Save menu option. The filename then is entered.

As this example illustrates, STACKEY is a great deal more powerful than ECHO.

## Just be quiet

`@echo off` is a great way to stop all the screen noise that occurs when batch file commands are being processed, but it doesn't stop the commands from displaying unwanted messages. If you try to rename a file and a problem occurs, for example, DOS displays the following message:

```
Invalid filename or file not found
```

To suppress these sorts of unwanted messages, you can use the following syntax to pipe the output of the command to the NUL device:

*command* `> nul`

The following commands are an extract from the RN.BAT batch file for renaming files:

```
....
5  if not exist %1 goto nosource
6  if exist %2 goto oldtarget
7  rename %1 %2 > nul
8  if not exist %2 goto failed
9  echo File Renamed!...
```

Notice that, because any messages the RENAME command on line 7 generates are redirected to the NUL device, the user doesn't see them. Refer to the IF command in Part IV of this book for the full RN.BAT listing.

To pause a batch file without displaying the standard message to press any key to continue, you can display your own message with ECHO and redirect the pause output to the NUL device:

```
@echo off
echo Press a key and we'll get busy....
pause > nul
```

Another way to suppress messages is to use the CTTY command to effectively turn off the screen and the keyboard. First, you execute the command

```
ctty nul
```

to redirect all I/O to the NUL device. You resume normal operations by executing this command:

```
ctty con
```

The following batch file skeleton shows this format:

```
@echo off
ctty nul
.....
.....
ctty con
```

This approach prevents having to redirect every individual command to the NUL device. Because the CTTY NUL statement effectively disables the keyboard, you should use this technique only when you are confident that the batch file is working and can handle all error conditions. The batch file ignores Ctrl-Break and any other key while CTTY is set to NUL.

## Displaying command output selectively

You learned in the preceding section how to suppress all output from a command by redirecting the output to a NUL device. If you want some, but not all, of the output, you can pipe the output to FIND and limit the output to the line that contains some specified text.

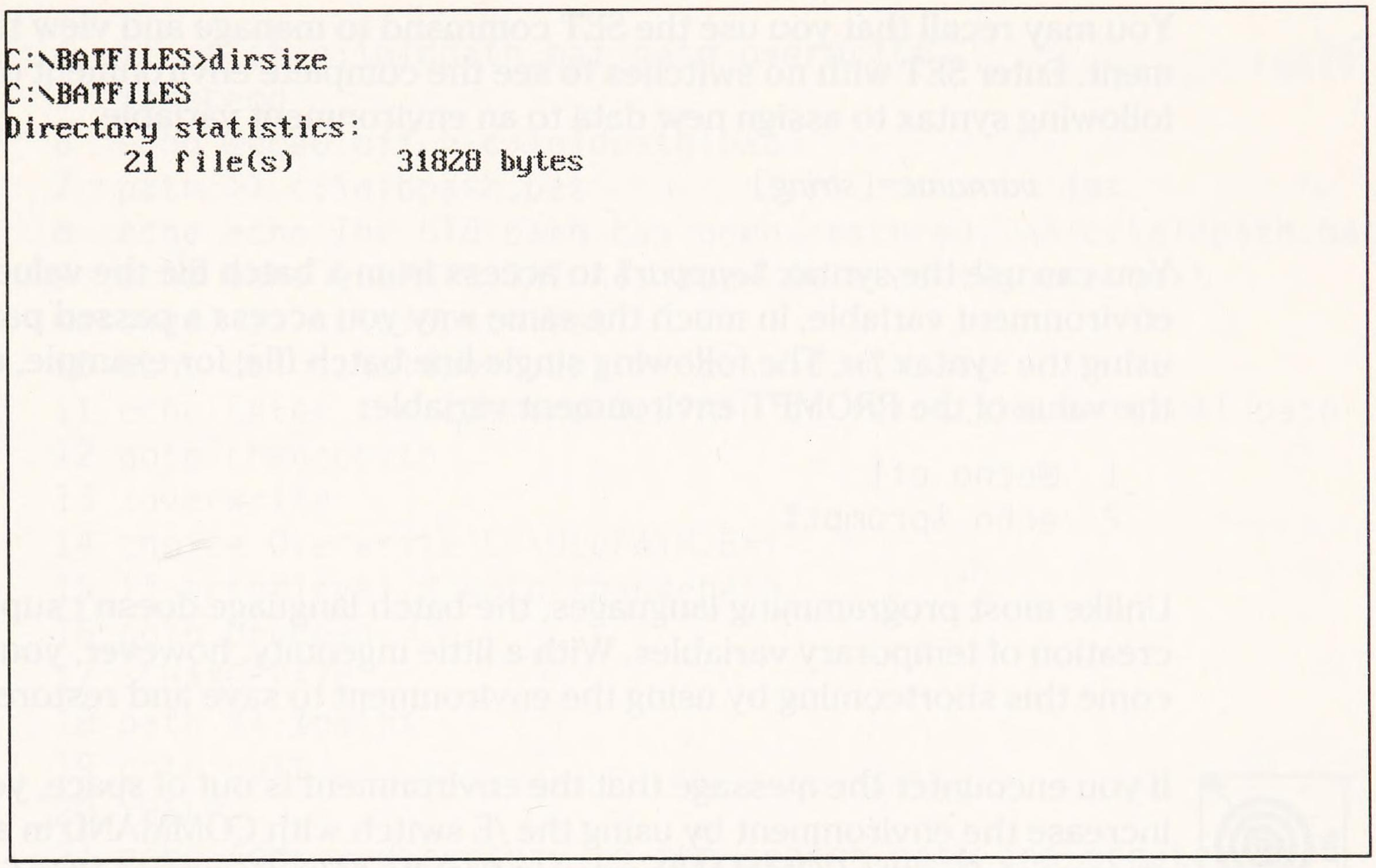
You can use this technique with the DIR command to show the number of files and space a directory uses. When you enter the DIR command with no switches, the display of volume details normally is followed by a list of all the files and then a summary. The word file(s) is always printed on the line that summarizes the directory statistics. You can display this single line by searching for the “file(s)” with FIND using the batch file DIRSIZE.BAT as follows:

```
@echo off
cd
echo Directory statistics:
dir | find "file(s)"
```

Figure 10-3 shows an example of the output this simple batch file generates.

The CHKFRAG.BAT batch file uses the same approach to filter the output from the following command, which reports individual file fragmentation:

```
chkdsk filemask
```



```
C:\BATFILES>dirsize
C:\BATFILES
Directory statistics:
     21 file(s)      31828 bytes
```

**Figure 10-3:** You can pipe command output to FIND to enable one specific line of output to be displayed.

After checking the integrity of the drive, CHKDSK reports individual file fragmentation or reports that there was no fragmentation. In either case, the word `contiguous` is displayed so that it can be the subject of the FIND command in the following CHKFRAG.BAT batch file:

```
@echo off
rem CHKFRAG checks for file fragmentation
if "%1"=="" goto CHKALL
echo Checking the file(s) %1 for fragmentation ...
chkdsk %1 | find "contiguous"
goto QUIT
:CHKALL
echo Checking all files in current directory for fragmentation ...
chkdsk *.* | find "contiguous"
:QUIT
```

## Environment Innovations

The DOS environment is a small storehouse for keeping general system settings that DOS and applications software use. The environment is also a useful resource for batch file programmers.

You may recall that you use the SET command to manage and view the environment. Enter SET with no switches to see the complete environment or use the following syntax to assign new data to an environment variable:

```
set varname=[string]
```

You can use the syntax `%envvar%` to access from a batch file the value of any environment variable, in much the same way you access a passed parameter by using the syntax `%n`. The following single-line batch file, for example, displays the value of the PROMPT environment variable:

```
@echo off
echo %prompt%
```

Unlike most programming languages, the batch language doesn't support the creation of temporary variables. With a little ingenuity, however, you can overcome this shortcoming by using the environment to save and restore data.

If you encounter the message that the environment is out of space, you can increase the environment by using the /E switch with COMMAND in a SHELL statement. See the COMMAND entry in Part IV, "The Complete DOS 6 Command Reference," for more information.

## Temporarily changing the search path

The PATH command does not have a switch for adding a directory to the search path. You can change PATH by editing the PATH statement in AUTOEXEC.BAT and reboot the system. Because the search path is stored in the environment, however, the following simple batch file, NEWPATH.BAT, makes the task easy:

```
1  @echo off
2  path %1;%path%
```

To add a new directory to the search path, run NEWPATH.BAT and pass the new directory as a parameter (NEWPATH C:\WP, for example). Line 2 sets the path to the passed parameter, followed by a semicolon and %path% (the current path).

You can refine the batch file by using the redirection characters > and >> to create another batch file, OLDPATH.BAT, which resets the path back to its original state:

```
1  @echo off
2  if "%1"=="" goto HELP
3  if "%1"=="/?" goto HELP
```

```
4 if exist c:\oldpath.bat goto overwrite
5 :proceed
6 echo @echo off > c:\oldpath.bat
7 path >> c:\oldpath.bat
8 echo echo The old path has been restored. >> c:\oldpath.bat
9 echo echo OLDPATH.BAT has been deleted (ignore next
message) >> c:\oldpath.bat
10 echo del c:\oldpath.bat >> c:\oldpath.bat
11 echo Enter the command OLDPATH to reset the original path
12 goto changepath
13 :overwrite
14 choice Overwrite C:\OLDPATH.BAT
15 if errorlevel 2 goto changepath
16 goto proceed
17 :CHANGEPATH
18 path %1;%path%
19 goto quit
20 :HELP
21 echo Adds a directory to the search path and creates an
OLDPATH batch file.
22 echo SYNTAX:     %1 pathdir
23 :quit
```

Lines 2 and 3 check to see whether the user failed to pass a parameter or passed the /? parameter. In either case, control jumps to line 21, which displays a brief help message.

The NEWPATH batch file dynamically creates a batch file called OLDPATH.BAT in the root directory of drive C. Line 4 uses an IF statement to determine whether the file already exists; if it does, the CHOICE command on line 14 prompts the user for confirmation before overwriting it. Line 6 uses the > redirector to create the new batch file (or overwrite the existing one) with the single line `@echo off` as the contents. Lines 7 through 10 use the >> redirector to add four more lines to the batch file. When you execute the temporary batch file OLDPATH, the path changes back to the original setting and the batch file deletes itself.

Finally, line 18 is executed to insert the %1 variable at the beginning of the path.

NOTE

The message that the batch file is missing is displayed when you execute OLDPATH. This message appears because the batch file deletes itself while DOS is executing it.

## Managing the TEMP variable

More and more programs, including Windows, are using the TEMP environment variable to ascertain where temporary files should be created. You can use the following simple batch file to assign a TEMP directory if one doesn't already exist or to prompt before changing an existing value:

```
@echo off
if "%temp%"=="" goto empty
choice Temp is currently set to %temp%. Overwrite it
if errorlevel 2 goto quit
:EMPTY
set temp=%1
Echo The TEMP environment variable is now %temp%
:QUIT
```

## Saving and restoring the environment

Bob's Shareware Collection disks include a useful batch file utility called PREFIX.EXE. PREFIX reads lines from one ASCII file and writes them to another but prefixes every line with a specified command. The syntax of PREFIX is

`prefix` *sourcefilename targetfilename prefixcommand*

PREFIX sets the following ERRORLEVELs:

0 Completed successfully

1 Not enough parameters specified

2 Could not open source file

3 Could not open (or create) target file

The following batch file, SAVEENV.BAT, uses PREFIX to create a temporary batch file called OLDENV for restoring the environment:

```
1  @echo off
2  set > temp.$$$
3  echo @echo off > c:\oldenv.bat
4  prefix temp.$$$ c:\oldenv.bat set
5  del temp.$$$ > nul
6  echo del c:\oldenv.bat >> c:\oldenv.bat
```

Line 2 pipes the output from the SET command to a temporary file, TEMP.$$$. Line 3 creates in the root directory of drive C a new batch file called OLDENV.BAT and adds the line `@echo off` to it. Line 4 uses PREFIX to read each line from TEMP.$$$ and add each line, with the prefix `"set "`, to the end of the file OLDENV.BAT. The file TEMP.$$$ is deleted, and the following line is added to the end of the OLDENV.BAT file:

```
del c:\oldenv.bat
```

After you run SAVEENV.BAT to save the environment settings, you can run OLDENV to restore the old environment.

You can use PREFIX in a variety of batch file situations. See the section "Saving the current directory," later in this chapter, for another example of PREFIX in action.

## Uppercasing input

When you use the syntax `%envvar%` to access an environment variable from a batch file, DOS uses a case-insensitive search of the environment to determine whether the variable exists. This capability provides a sneaky way to make a case-insensitive test of parameters passed to batch files.

The following batch file fragment uses traditional techniques to check for both upper- and lowercase input:

```
....
if "%1"=="RED" goto red
if "%1"=="red" goto red
....
```

Unfortunately, if you enter `red` in another case mixture (`Red`, for example), the batch file doesn't recognize it. Although you can test for every possible combination of upper- and lowercase letters for a short word like *red,* testing on a longer word, such as *lightmagenta,* is totally impractical.

If you use the passed parameter to create an environment variable, you can then use the *%envvar%* syntax in an IF statement to perform a case-insensitive test. The following batch file fragment uses this technique:

```
....
set $$%1=clever
if "%$$red%"=="clever" goto RED:
if "%$$blue%"=="clever" goto BLUE:
....
 set %1=
....
```

The passed parameter is prefixed with $$ before it is added to the environment. This quick and easy method reduces the chances of overwriting an existing environment variable — I've never encountered any variable that normally begins with $$.

In the example, the text *clever* was assigned to the environment variable, and the same text is used in the IF statements. You can use any text for this purpose, but keep it short to reduce environment space consumption.

The IF statement on line 5 intercepts a parameter of `red`, regardless of whether it is `RED`, `Red`, `red`, `ReD`, or any case mixture.

The only drawback with this technique is that, because it uses the environment, the environment can run out of space.

An alternative is to echo the passed parameter and pipe it to the FIND command. By specifying the /I switch with FIND, you can perform a case-insensitive search. Here is an example:

```
echo %1 | find "red" /i
```

This command needs a few refinements. First, if FIND locates the text, it normally echoes the text to the screen, but this capability is not useful in this application. If FIND does not locate the text, however, a nonzero ERRORLEVEL is set. By piping the FIND output to the NUL device and testing for a nonzero ERRORLEVEL, you can implement a case-insensitive test:

```
echo %1 | find "red" /i > nul
if not errorlevel 1 echo You entered RED!
```

One more refinement is required. FIND searches for the characters red anywhere in the input parameter string. The preceding test responds in the same way to `red` and `lightred`, or any other string containing the substring `red`. By inserting the search string between two characters, you can ensure that there is a complete match on the full passed parameter. You can surround the string parameter in the example code with two $ characters:

```
echo $%1$ | find "$red$" /i > nul
if not errorlevel 1 echo You entered RED!
```

DOS 6

## Enhancements to FIND

One of the more subtle enhancements to DOS 6 is the addition of ERRORLEVEL support in the FIND command. The improved FIND command returns the following ERRORLEVELs indicating whether or not the text was found:

| | |
|---|---|
| ERRORLEVEL 0 | The text was found. |
| ERRORLEVEL 1 | The text was not found. |
| ERRORLEVEL 2 | An error occurred and the search failed. |

Although this may seem like a very minor enhancement, it provides a whole grab bag of new batch file techniques. Many of the batch files in the remainder of this chapter exploit the new ERRORLEVEL support in FIND.

# Dates, Times, and Batch Files

On the surface, batch files seem to have no support for date- and time-related tasks; lurking beneath the surface, however, are a number of techniques you can use.

## Waiting for a while

Many programs pause to display a message or logo for a brief time and then start executing tasks. You can implement the same facility in a batch file by using the CHOICE command to implement a fixed-period delay.

By using the /N switch to suppress the `[Y,N]`-style prompt, and the /T switch to specify the default key and the delay time, you suppress all output from CHOICE and make it wait a specified number of seconds. The following command pauses for five seconds and then proceeds with the next statement in the batch file:

```
choice /c:a /n /t:a,5
```

The only weakness with this approach is that the time-out period is infinite if the user presses an invalid key. To work around the problem, use the /C switch to specify all the standard keys. Then, if the user presses a key, the worst that can happen is that CHOICE will finish early and the user will continue — most users would probably see this as a useful feature.

**NOTE** If the user presses a nonlisted key (such as the spacebar or Enter), the batch file pauses until the user presses a standard key or Ctrl-Break.

The following batch file uses this technique to display a copyright notice:

```
@echo off
cls
echo.
echo.
echo.
echo      ╔════════════════════════════════════════════╗
echo      ║                                            ║
echo      ║              Batch Files R Us              ║
echo      ║                   v2.05                    ║
echo      ║                                            ║
echo      ║    Copyright 1993 TechnoJock Software, Inc. ║
echo      ║                                            ║
echo      ╚════════════════════════════════════════════╝
choice /c:abcdefghijklmnopqrstuvwxyz12345678790 /n /t:A,5
cls
echo Lets get busy...
....
```

## What day of the week is it?

The tiny program WEEKDAY.COM sets an ERRORLEVEL to indicate the day of the week. Sunday is represented by an ERRORLEVEL of 0, Monday by 1, and through to Saturday, with an ERRORLEVEL of 6. WEEKDAY.COM, an 8-byte (count 'em) program created in a few seconds with DEBUG, is available on Bob's Shareware Collection disks. Chapter 11 explains how to use DEBUG to create WEEKDAY.COM, as well as WHATDAY.COM, WHATMON.COM, and WHATYEAR.COM.

The following skeleton batch file, TODAYIS.BAT, shows how to test the ERRORLEVEL returned by WEEKDAY.COM to determine the day of the week:

```
@echo off
weekday
if errorlevel 6 echo Saturday
if errorlevel 5 if not errorlevel 6 echo Friday
if errorlevel 4 if not errorlevel 5 echo Thursday
if errorlevel 3 if not errorlevel 4 echo Wednesday
if errorlevel 2 if not errorlevel 3 echo Tuesday
if errorlevel 1 if not errorlevel 2 echo Monday
if errorlevel 0 if not errorlevel 1 echo Sunday
```

## What time is it?

Even a simple task like displaying the current time involves a few tricks. As you know, you use the TIME command to modify the system time. When you enter the TIME command, output similar to the following is displayed:

```
Current time is 12:00:11.86a
Enter new time:
```

Although TIME displays the current time, it waits for you to change the time or press Enter. Because ECHO commands always include a carriage return, you can use the ECHO command to feed an Enter to the TIME command:

```
echo.|time
```

The `echo.` statement instructs ECHO to draw a blank line and then enter a carriage return. This carriage return is piped to TIME to emulate your pressing Enter.

### Checking ERRORLEVELs in Batch Files

The IF statement is used in batch files to test the ERRORLEVEL returned by a program. One quirk of an IF ERRORLEVEL test is that a TRUE value is applied if the ERRORLEVEL is equal to *or greater than* the test value. The important point to note is that a larger ERRORLEVEL returns a TRUE value in an IF ERRORLEVEL test. For example, if the actual ERRORLEVEL returned by a program is 2, the following expression will always ECHO true:

```
if errorlevel 1 echo true
```

One way to accommodate this characteristic of the IF ERRORLEVEL test is to test ERRORLEVELs in descending order and then branch to another location of the batch file using GOTO. The following batch file fragment illustrates this technique:

```
if errorlevel 2 goto label1

if errorlevel 1 goto label2

if errorlevel 0 goto label3
```

If you do not use GOTO in the IF ERRORLEVEL statement to branch to another location in the batch file, DOS proceeds with the next IF statement on the subsequent line. Each subsequent IF ERRORLEVEL test would also respond TRUE. If you don't use a GOTO statement, you should use two IF ERRORLEVEL statements in a single expression — one testing for a value and the other testing the next larger value. Here is an example:

```
if errorlevel 2 if not errorlevel 3 echo 2

if errorlevel 1 if not errorlevel 2 echo 1

if errorlevel 0 if not errorlevel 1 echo 0
```

The batch file TODAYIS.BAT shown in the preceding section uses this technique.

The revised TIME command still displays two lines of text, but you can filter out the second line with the FIND command. By searching for the word `Current`, FIND displays only the first line of the text. The complete expression for displaying the current time, therefore, becomes

```
echo.|time|find "Current"
```

When you execute this command, output similar to the following is displayed:

```
Current time is 12:00:11.15a
```

## Waiting for a specific time

By using the technique outlined in the previous section, you can use the TIME command to pause a batch file until a specific time is reached. The following batch file, WAITFOR.BAT, waits until a specified time is reached before executing one or more commands:

```
1  @echo off
2  if "%1"=="" goto help
3  if "%1"=="/?" goto help
4  set $$$$TIME=is %1
5  echo Waiting for time %1
6  :LOOP
7  echo.|time|find "%$$$$TIME%" > nul
8  if errorlevel 0 if not errorlevel 1 goto ontime
9  goto loop
10 :ONTIME
11 set $$$$TIME=
12 rem Insert the timed commands below
13 echo Alarm Alarm Alarm!!!
14 goto quit
15 :HELP
16 echo Pauses batch file execution until a specific time
17 is reached.
18 echo SYNTAX %0 HH:MM[:SS], e.g. %0 11:31
19 :QUITS
```

The batch file should be passed a single parameter that identifies the start time. The time can be passed in a format similar to the TIME command, to specify hours:minutes and, optionally, seconds.

Line 4 creates a temporary environment variable called `$$$$TIME`, which is set to the characters `"is "` followed by the user-specified time. The `"is "` prefix is used to ensure that FIND does not report a match if the minutes and seconds coincidentally equal the target hours and minutes. If the desired time was 10:20, for example, and TIME returned the following value, a match for `is 10:20` would not be found:

```
Current time is  9:10:20.04a
```

A simple search for `10:20`, however, would have erroneously returned a match.

On line 7, the output from TIME is passed to FIND, which searches for the environment variable string. This test is repeated continuously until FIND returns an ERRORLEVEL of 0, to indicate that the time has arrived. The batch file then branches to line 11, in which the environment variable is removed and the appropriate time-sensitive commands are executed.

## Running batch files once a day

I like to run some programs once a day, such as a backup of some key files and a virus scan. The problem is that I reboot my system ten or more times a day, and I want to execute these commands once every day — not once every reboot. I cannot, therefore, add the commands to the AUTOEXEC.BAT file. Granted, I can add CHOICE commands to AUTOEXEC.BAT to prompt whether programs should be executed, but even responding to just two or three prompts is irritating. I can create a once-a-day batch file, but — as usual — some tricks must be used. Many of the techniques used to pause a batch file until a specified time (discussed in the preceding section) are deployed in this situation.

The key to solving the problem is to store today's date in an environment variable. The following batch file, SETDATE.BAT, stores the date in the environment variable TODAYIS:

```
1  echo off
2  echo.|date|find "Current"> $$$$DATE
3  echo @echo off > $$$1date.bat
4  echo set TODAYIS=%%5>> $$$1date.bat
5  echo @echo off > $$$2date.bat
6  prefix $$$$DATE $$$2date.bat $$$1date
7  call $$$2date
8  del $$$$DATE
9  del $$$1DATE.BAT
10 del $$$2DATE.BAT
11 :quit
```

This batch file creates three temporary files. The first file, $$$$DATE, is created on line 2 and contains the output from the first line of the DATE command, as shown in this example:

```
Current date is Sun 01-03-1993
```

Lines 3 and 4 create a second file, $$$1DATE.BAT, used to create the environment variable TODAYIS. $$$1DATE.BAT has the following contents:

```
@echo off
set TODAYIS=%5
```

The third file, $$$2DATE.BAT, is created by using the PREFIX command (included on Bob's Shareware Collection disks) by prefixing the contents of $$$$DATE with the word $$$1DATE. $$$2DATE.BAT has the following contents:

```
@echo off
$$$1date Current date is Sun 01-03-1993
```

This batch file calls the $$$1DATE batch file and passes five passed parameters. Notice that the fifth passed parameter is the system date. $$$1DATE in turn creates the environment variable TODAYIS to assign the fifth passed parameter as the contents. Finally, lines 8 through 10 delete the three temporary files. Mission accomplished.

By calling SETDATE, you can store the system date in the environment. The following batch file, ONLYONCE.BAT, uses the environment variable to execute one or more commands once a day:

```
1  @echo off
2  call setdate
3  if exist c:\%todayis%.oo$ goto quit
4  if exist c:\*.oo$ del c:\*.oo$
5  echo nul > c:\%todayis%.oo$
6  rem Insert Once a Day commands Below here
7  echo Good morning
8  :QUIT
```

ONLYONCE creates an empty file that uses the first eight characters of today's date (grabbed from the environment) for the name, with the unique extension OO$. Line 3 checks to see whether the file exists; if so, the batch file has already been run once today and therefore quits. If the file doesn't exist, all files with the OO$ extension are deleted to remove the previous day's (or days') files (see line 4), and an empty file is created by piping an ECHO NUL statement to the new file (see line 5). In this example, the text `Good morning` is displayed; in a real application, however, the once-a-day commands are inserted at line 7.

## A Batch File Potpourri

All good programmers have a collection (or *library*) of routines in their code arsenal. This section lists a bevy of batch file routines you can add to your library.

### Does a directory exist?

You can easily use the IF command to determine whether a file exists. The following extract from a batch file shows IF checking whether the file CLOCK.WCH exists:

```
....
if exist CLOCK.WCH goto NextStep
goto Error3
....
```

Checking for a directory is slightly trickier. The following batch file echoes `Found` only if at least one file is in the directory (excluding the . and .. "files"):

```
@echo off
if exists c:\empty\*.* echo Found
```

If no files are in the directory, the IF test fails. To determine whether a directory exists, even when it is empty, you must use IF to test for the NUL device by using the following syntax:

`if exist [`***drive***`:]\`*pathname*`\nul` *command*

The following batch file accepts a single path as a parameter and then creates and changes to the new directory:

```
1  @echo off
2  if "%1"=="" goto HELP
3  if exist %1\nul goto OK
4  md %1 > nul
5  if exist %1\nul goto OK
6  echo The directory could not be created
7  goto quit
8  :help
9  echo Syntax is %0 drive:\newpath
10 goto QUIT
11 :OK
12 cd %1
13 :QUIT
```

Line 3 determines whether the directory exists by testing the NUL device. If the directory exists, processing jumps to line 12. If it doesn't exist, an MD command is executed and the test is repeated. If the directory still doesn't exist, the directory name must have had an error. If all is well, line 12 changes the default directory to the new directory.

## Checking for wildcards

The FOR command provides an easy way to determine whether a filename contains wildcards. When a wildcard is used in a FOR statement, a different specific filename is used with every iteration of the loop. If the substituted filename used in the FOR loop is exactly the same as the filename being tested, the filename being tested does not contain wildcard characters. The following batch file, TESTWILD.BAT, shows this technique in action:

```
@echo off
if "%1"=="" goto NOTWILD
for %%W in (%1) do if %%W==%1 goto NOTWILD
```

```
:ISWILD
echo Wildcard Entered
goto quit
:NOTWILD
echo No wildcard Entered
:QUIT
```

## Checking the DOS version

If you are writing a batch file that uses new batch or DOS commands, you may want to test to ensure that the appropriate version of DOS is being used. Any batch file that uses the new CHOICE command, for example, runs only on DOS 6 systems.

The FIND command was enhanced in DOS 6 to return an ERRORLEVEL indicating whether or not the search text was found. You may recall that an ERRORLEVEL of 1 is always returned, when the text is not found. Since the FIND command for earlier versions of DOS would always return an ERRORLEVEL of zero, you can test for the presence of DOS 6 by searching for some text you know does not exist. If an ERRORLEVEL of 1 is returned, DOS 6 (or later) is being used; otherwise, an earlier version of DOS is in use.

The following example batch file, ISDOS6.BAT, shows how to use VER with FIND to search for some nonexistent text and thereby test for a specific version of DOS:

```
1  @echo off
2  ver | find "Lloyd Belcher" > nul
3  if errorlevel 1 goto OK
4  echo This Batch file requires DOS 6
5  goto quit
6  :OK
7  echo Running DOS 6
8  :QUIT
```

Line 3 sends (or pipes) the output from the VER command to FIND, which searches for the string `Lloyd Belcher`. (I promised I would mention Lloyd somewhere in the book!) You could search for any text that is not output from VER. Line 3 tests the ERRORLEVEL and if it returns 1, DOS 6 must be in use.

## Checking for TSRs

DOS 6 introduced the /M (or /MODULE) switch for the MEM command. This switch instructs MEM to display the status of a specific program in memory. If the program is not already loaded, MEM displays this string:

*progname* `is not currently in memory.`

By piping the output from MEM /M to FIND and testing for the string `not currently`, a batch file can easily test to see whether a program is loaded in memory.

The following batch file uses this technique to determine whether the user-specified program is loaded in memory:

```
@echo off
if "%1"=="" goto help
if "%1"=="/?" goto help
mem /m %1|find "not currently" > nul
if errorlevel 1 goto loaded
:NOTLOADED
echo %1 is not in memory
goto quit
:LOADED
echo %1 is already loaded
goto quit
:HELP
echo Checks to see if a program is loaded in memory
echo SYNTAX %0 progname, e.g. %0 share
:QUIT
```

By using this simple technique, you can test for any memory-resident program, including DOSKEY, SHARE, EMM386, and even Windows (discussed later, in the section "Using Batch Files with Windows 3.1").

To determine the exact program name in memory, use the command MEM /D /P.

## Saving the current directory

I don't know about you, but I am always changing from one directory to another and then soon after want to return to the original directory. Again, a batch file can simplify the task.

The following batch file, TCD.BAT (for *t*emporary *c*hange *d*irectory), changes directories and uses PREFIX.EXE to create a second batch file, CB.BAT (for *c*hange *b*ack):

```
@echo off
if "%1"=="" goto help
if "%1"=="/?" goto help
if exist c:\cb.bat goto overwrite
:PROCEED
echo.
echo Enter CB to return to the directory
```

```
8 cd
9 echo.
10 echo @echo off > c:\cb.bat
11 cd > $$$$TEMP
12 prefix $$$$TEMP c:\cb.bat CD
13 del $$$$TEMP
14 echo del c:\cb.bat >> c:\cb.bat
15 goto CHANGEDIR
16 :OVERWRITE
17 choice Overwrite C:\CB.BAT
18 if errorlevel 2 goto changedir
19 goto proceed
20 :CHANGEDIR
21 cd %1
22 goto quit
23 :HELP
24 echo Changes directory and creates a CB batch file to
change back.
25 echo SYNTAX: %0 newdir, e.g. %0 \DOS
26 :QUIT
```

To change to a new directory, you simply enter TCD and the directory name. The batch file then creates a new batch file, CB.BAT, which contains the commands to change back to the original directory and delete itself (see lines 10 through 14).

# Using Batch Files with Windows 3.1

Many DOS 6 users are also Windows 3.1 users. This section discusses some batch file considerations for running batch files from within Windows 3.1.

## Controlling DOS session settings

When you start a DOS session, Windows normally displays a banner to remind you that you are running under Windows and describes some useful key combinations, such as Exit, Alt-Tab, and Alt-Enter. To suppress this default message, edit the SYSTEM.INI file so that the following statement is located in the [386Enh] section:

```
[386Enh]
DOSPromptExitInstruc=false
```

By adding a WINPMT environment variable before starting Windows, you can define a special prompt that is active in every DOS session. The following command in AUTOEXEC.BAT defines a special Window's prompt:

```
set winpmt=Running within Windows!$_$p$g
```

Windows 3.1 allocates by default the same environment space as defined in CONFIG.SYS. To instruct Windows to allocate a different-size environment, add a CommandEnvSize= statement to the [NonWindowsApp] section of SYSTEM.INI. The following extract from SYSTEM.INI creates an 800-byte environment in every DOS session:

```
[NonWindowsApp]
CommandEnvSize=800
```

## Exploiting WINSTART.BAT

Whenever you execute WIN, Windows looks for the file WINSTART.BAT in the WINDOWS directory. Windows executes this file before shifting into graphics mode and launching the Windows shell.

The primary purpose of this file is to load TSRs that Windows requires but that are not needed by any DOS session running within Windows. All TSRs loaded before you launch Windows are installed automatically in every DOS session. By loading TSRs from WINSTART.BAT, you save memory for every DOS session. Not many TSRs are designed just for Windows, but they are usually related to networks or communications.

The standard WINSTART.BAT batch file can include any commands. On one system, I use the following commands to remind the user to enter a Bernoulli cartridge before proceeding:

```
@echo off
echo Make sure a cartridge is installed in drive D
pause
```

You can use the file also to copy the system files WIN.INI and SYSTEM.INI, just in case a pesky program decides to modify them on the fly:

```
@echo off
echo Copying system files
copy c:\windows\win.ini c:\windows\oldwin.ini
copy c:\windows\system.ini c:\windows\oldsys.ini
echo Loading Windows...
```

## Using the new command /K switch

When you launch the MS-DOS icon, Windows executes the file DOSPRMPT.PIF. Normally, this PIF file executes COMMAND.COM to start a DOS session.

DOS 6 introduced the new /K switch for COMMAND.COM, which enables you to specify the name of a batch file or program to be executed when COMMAND.COM is launched. This switch provides a convenient way to customize the DOS session. It's like having a custom AUTOEXEC.BAT for every DOS session.

To take advantage of this new switch, run PIFEDIT (from within Windows) and load the DOSPRMPT.PIF file from the Windows directory. On the Optional Parameters line, add a /K followed by a space and the name of a batch file. The batch file will be executed every time you start a DOS session.

This sample batch file changes the prompt and shows the amount of available memory:

```
@echo off
prompt RUNNING IN WINDOWS$_$P$G
mem | find "executable"
```

## Checking for Windows 3.1

Many commands should not be run from a Windows DOS session (CHKDSK and DEFRAG, for example). You should avoid loading certain TSRs, such as disk caches and memory managers, in DOS sessions. Sometimes, therefore, you will want to know whether Windows is loaded (whether the batch file is being executed from a DOS session) before executing one or more commands.

You may have noticed that Windows automatically creates an environment variable, *windir*, in every DOS session. This variable is used to record the main Windows directory. Here's the crazy part: Windows creates the environment variable in *lowercase*. Because the command interpreter always makes environment variable names uppercase, a batch file has no way to test whether *windir* is present. Shucks.

Fortunately, you can easily test for Windows by using MEM with the new /M switch, as described in the section "Checking for TSRs," earlier in this chapter. The following batch file tests whether Windows is loaded in memory:

```
mem /m win|find "not currently" > nul
if errorlevel 1 goto WinLoad
echo Windows is not loaded
goto quit
:WINLOAD
echo Windows is loaded
:QUIT
```

# Batch File Enhancers

Bob's Shareware Collection disks include two programs designed to make batch files more efficient and powerful.

## Adding pizzazz with Batutil

Batutil, from CtrlAlt Associates, takes batch file programming capabilities to an entirely new level. You can create pop-up menus, add excellent sound effects, edit the environment interactively, write to the screen in full color (without ANSI.SYS), determine installed equipment, perform math, and easily manage dates and times.

The following small batch file creates the menu shown in Figure 10-4.

```
1  @echo off
2  batutil {AT 30}{CL}
3  batutil {AT 30}{BI 219}{BE $S$SBUSTER}
4  batutil {FK}{SH}{AT 0}{MH Main menu}{ME Incremental$SBackup
Full$SBackup Windows Quit}
```

BATUTIL uses two-character commands. Line 2 sets the display attribute to 30 (black on cyan) and clears the screen. Line 3 instructs BATUTIL to use *big* characters, and the command BE (for *b*atutil *e*cho) draws the string "  BUSTER" in big

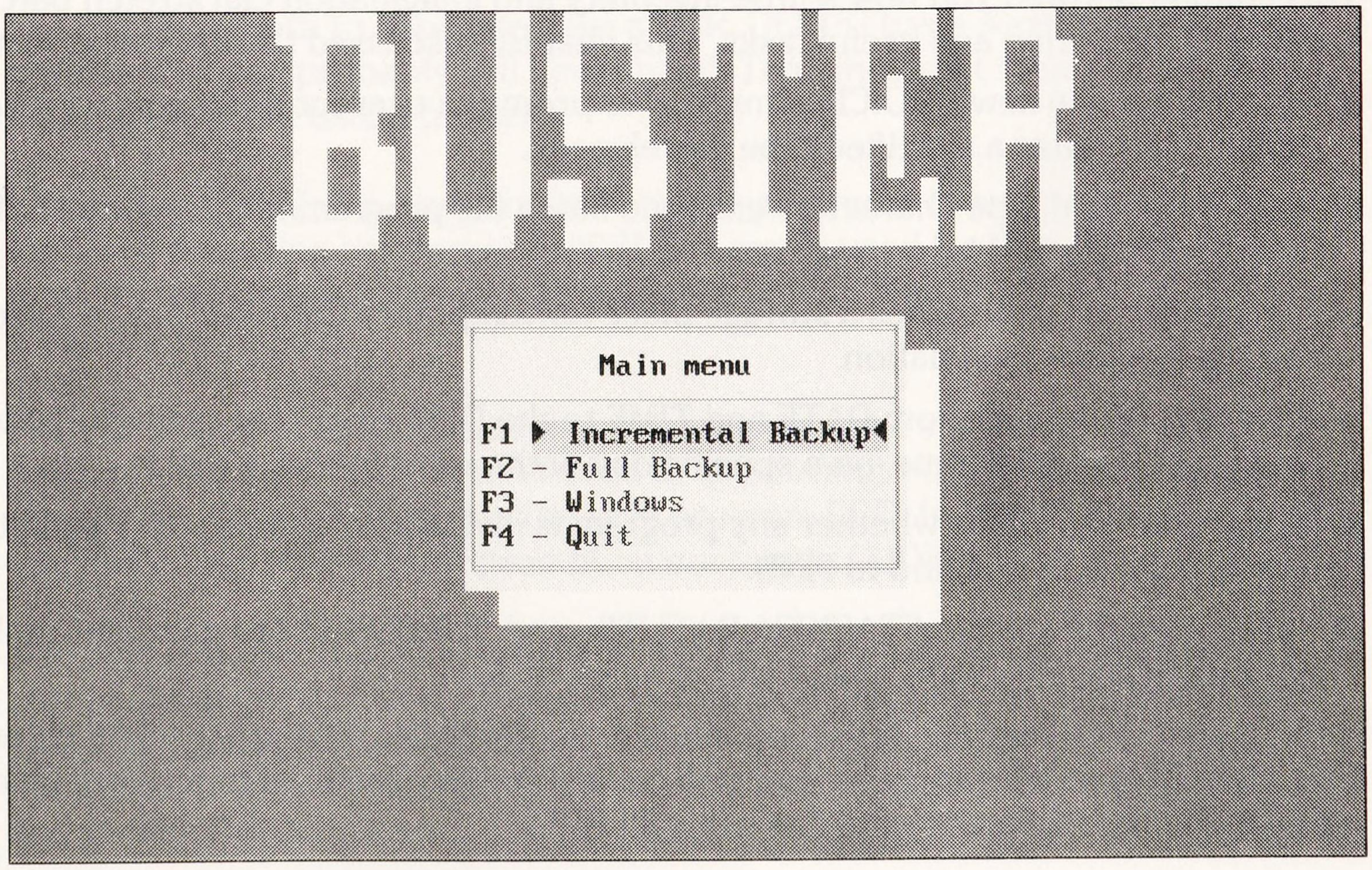


**Figure 10-4**: A fancy menu screen created with BATUTIL.

letters. The metastring $S denotes a space. Line 4 is responsible for displaying the menu. The commands {FK} and {SH} enable function-key support and shadows, respectively. The menu options are then defined with the {ME} command. In a real application, the ERRORLEVEL is tested to determine which option the user chose.

To gain a quick appreciation of the versatility of BATUTIL, run the BUDEMO.BAT file included with the Batutil shareware package.

## Putting batch files in overdrive with TURBOBAT

TURBOBAT, part of the ExtraDOS Toolbox from Foley Hi-Tech Systems, compiles batch files into executable programs (with a COM extension). This product not only prevents users from tampering with your batch files, but it also speeds execution speed as much as 400 percent. It is a must for every batch file writer.

As if those features aren't enough, TURBOBAT also extends the batch language to include a grab bag of useful enhancements, including full color and sound support, much-improved flow control with IF, REPEAT, and WHILE statements, the capability to save multicharacter user input to an environment variable, and many more. It is firmly entrenched in my DOS bag of tricks.

### *Summary*

This chapter showed you how a little ingenuity and imagination can stretch batch files to perform many varied and useful tasks. This chapter discussed the following points:

- You can use the new CHOICE command to prompt a user for input and optionally return a default value after a specified time has elapsed.
- Redirection and pipe characters enhance batch file programming by creating files and passing data to and from programs.
- The environment provides a storage place for temporary data and can be used to perform case-independent validation.
- By piping the output from DATE and TIME to the FIND filter, you can write batch files that execute at a specific time, on a specific day of the week, or once a day.
- A batch file can check whether any program is installed in memory by piping the output from a MEM /C command to FIND.
- The shareware utilities STACKEY, BATUTIL, and TURBOBAT are excellent tools for enhancing and extending the power of batch files.

Although you can manage DOS commands with batch files, it is not a very rich programming language. Chapter 11 describes how you can extend the power of batch files by creating compact high-performance utilities using DEBUG.

# Chapter 11

# Demystifying DEBUG

### *In This Chapter*

- Reviewing DEBUG's many uses
- Getting acquainted with DEBUG editing capabilities
- Using DEBUG commands
- Patching programs with DEBUG
- Creating useful utilities with DEBUG

## Understanding VCRs, Understanding DEBUG

Let's get one thing clear: DEBUG is useful.

I think VCRs and DEBUG have a lot in common. At first, both seem unnecessarily complicated and cumbersome, but after you learn the basic features, you can put them both to good use. Like a VCR, DEBUG has a wealth of advanced options that you probably will never need. That shouldn't stop you from exploiting the more useful features.

Ironically, DEBUG was designed to help programmers debug programs that were not operating as expected. DEBUG is rarely used for that purpose nowadays because far superior debuggers are available. But DEBUG can do a lot more than find bugs.

With DEBUG, you can edit existing programs to make small changes and enhancements, a process often referred to as *patching*. For example, you can change the default filemask in the DOS program EDIT from *.TXT to *.BAT, and you can change the default prompt characters in CHOICE.

You also can use DEBUG to create complete programs, but don't expect to write a three-dimensional spreadsheet with DEBUG. You can, however, create some very useful and compact utilities — for example, utilities to reboot the computer, to test whether Shift is held down, to determine the day of the week, and to turn the cursor off.

By the way, how do you record television shows from two different channels?

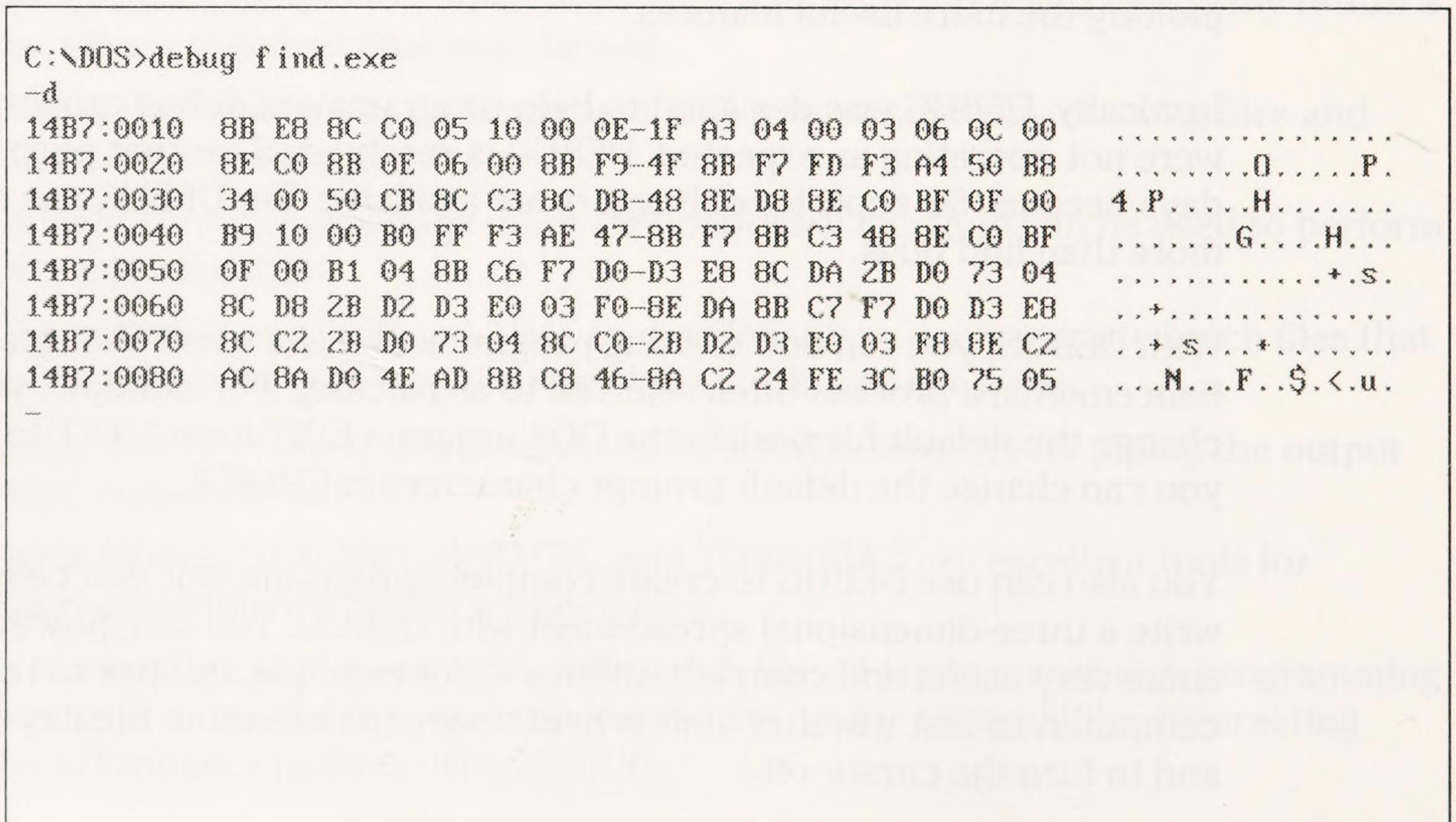
## Getting Acquainted with DEBUG Editing Capabilities

As you know, EDIT is used to edit ASCII files such as batch and INI files. DEBUG also is a file editor, but it is designed to edit binary program files — that is, files with a COM or EXE extension. Unlike ASCII files, binary files are not organized into individual lines; a binary file is one homogenous blob of bytes. DEBUG does not, therefore, have a line-number orientation. File contents are displayed in blocks of bytes.

Figure 11-1 shows a typical display generated with DEBUG's D (dump) command. The display is divided into three sections. The leftmost section identifies the address (using segment:offset notation) of the first byte of data, and the middle section shows the value of 16 bytes of the file in hexadecimal. The rightmost section shows the same 16 bytes of data in ASCII format. In that section, alphanumeric characters (that is, characters with ASCII values in the range 32 to 126) are displayed; other, nondisplayable characters are represented by the period (.) character.

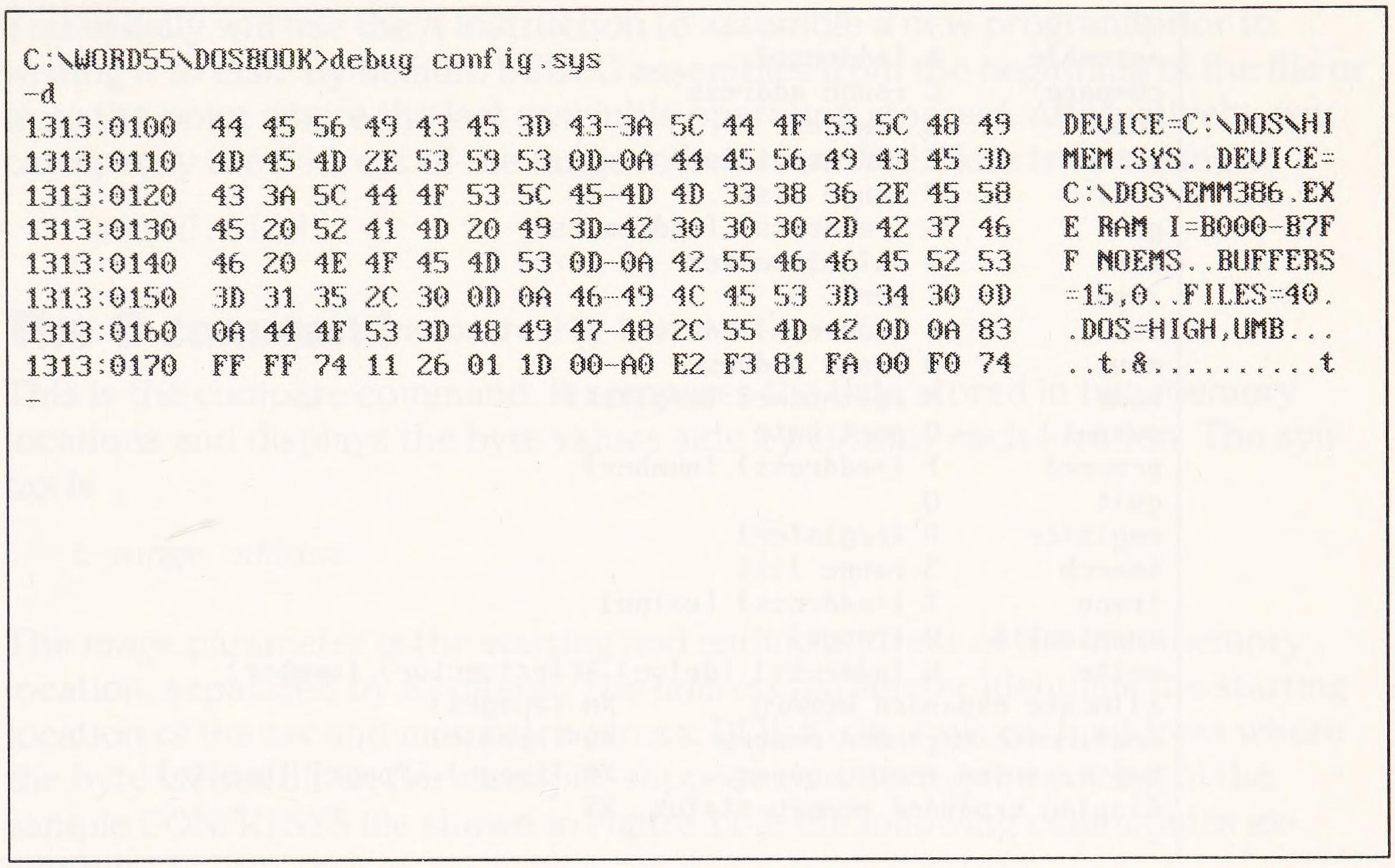
DEBUG can edit any type of file, not just binary ones. Consider the following barebones CONFIG.SYS file:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM I=B000-B7FF NOEMS
BUFFERS=15,0
FILES=40
DOS=HIGH,UMB
```



```
C:\DOS>debug find.exe
-d
14B7:0010  8B E8 8C C0 05 10 00 0E-1F A3 04 00 03 06 0C 00   ................
14B7:0020  8E C0 8B 0E 06 00 8B F9-4F 8B F7 FD F3 A4 50 B8   ........O.....P.
14B7:0030  34 00 50 CB 8C C3 8C D8-48 8E D8 8E C0 BF 0F 00   4.P.....H.......
14B7:0040  B9 10 00 B0 FF F3 AE 47-8B F7 8B C3 48 8E C0 BF   .......G....H...
14B7:0050  0F 00 B1 04 8B C6 F7 D0-D3 E8 8C DA 2B D0 73 04   ............+.s.
14B7:0060  8C D8 2B D2 D3 E0 03 F0-8E DA 8B C7 F7 D0 D3 E8   ..+.............
14B7:0070  8C C2 2B D0 73 04 8C C0-2B D2 D3 E0 03 F8 8E C2   ..+.s...+.......
14B7:0080  AC 8A D0 4E AD 8B C8 46-8A C2 24 FE 3C B0 75 05   ...N...F..$.<.u.
-
```

**Figure 11-1**: The D (dump) command displays file contents in three distinct blocks.



```
C:\WORD55\DOSBOOK>debug config.sys
-d
1313:0100  44 45 56 49 43 45 3D 43-3A 5C 44 4F 53 5C 48 49   DEVICE=C:\DOS\HI
1313:0110  4D 45 4D 2E 53 59 53 0D-0A 44 45 56 49 43 45 3D   MEM.SYS..DEVICE=
1313:0120  43 3A 5C 44 4F 53 5C 45-4D 4D 33 38 36 2E 45 58   C:\DOS\EMM386.EX
1313:0130  45 20 52 41 4D 20 49 3D-42 30 30 30 2D 42 37 46   E RAM I=B000-B7F
1313:0140  46 20 4E 4F 45 4D 53 0D-0A 42 55 46 46 45 52 53   F NOEMS..BUFFERS
1313:0150  3D 31 35 2C 30 0D 0A 46-49 4C 45 53 3D 34 30 0D   =15,0..FILES=40.
1313:0160  0A 44 4F 53 3D 48 49 47-48 2C 55 4D 42 0D 0A 83   .DOS=HIGH,UMB...
1313:0170  FF FF 74 11 26 01 1D 00-A0 E2 F3 81 FA 00 F0 74   ..t.&..........t
-
```

**Figure 11-2:** You can use DEBUG to view and edit ASCII files.

Figure 11-2 shows a DEBUG dump of this file. Notice that the end of the first text line (shown in the middle of the second line of the dump) is followed by the two characters *0D* and *0A*. These characters are carriage return and line feed, respectively, and are used behind the scenes in ASCII files to identify the end of a new line of text.

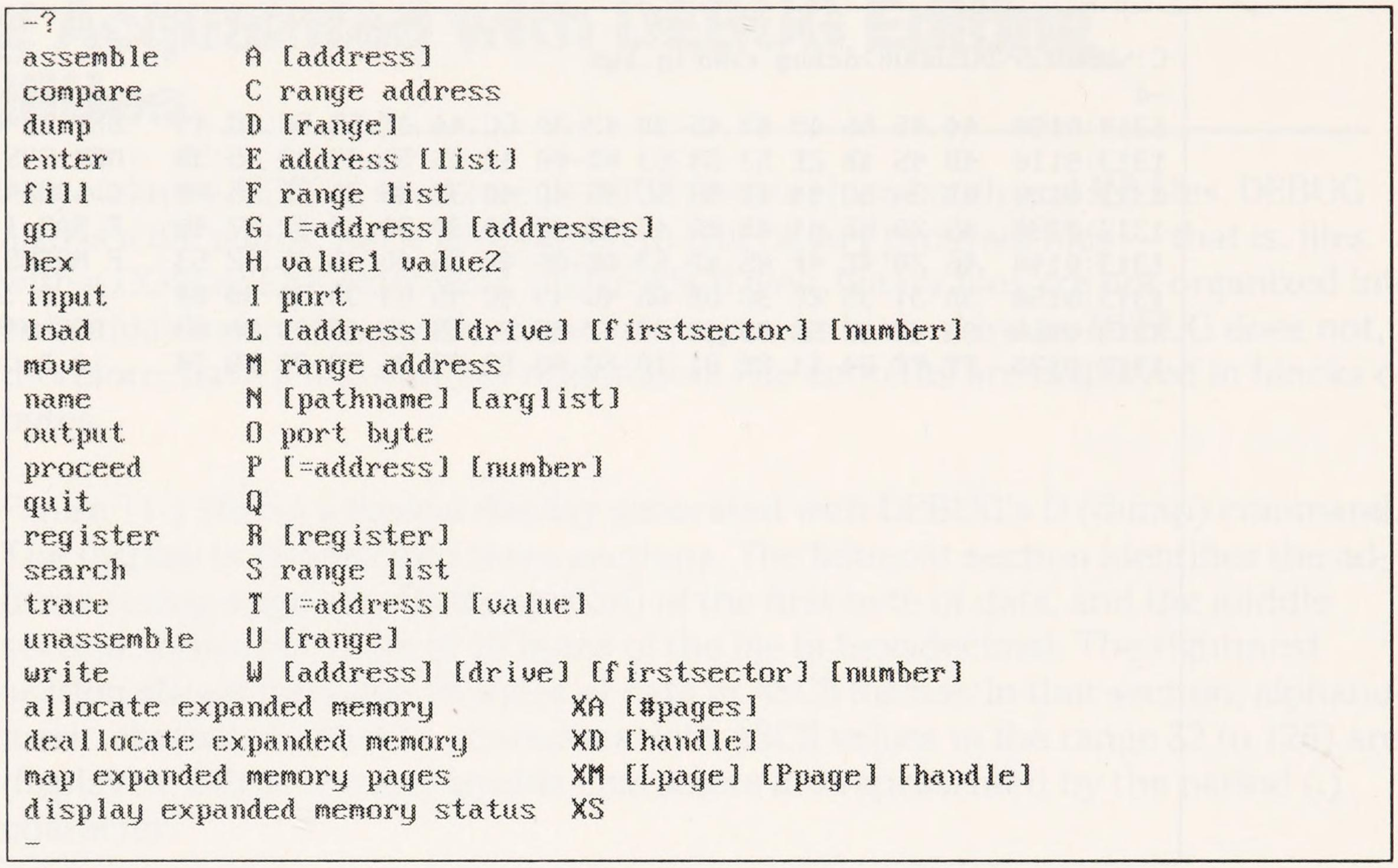
## Using DEBUG Commands

When you execute DEBUG, a single character is displayed: the minus sign ( – ). This character is the terse DEBUG prompt that indicates that DEBUG is waiting for you to input a command or instruction. It's not the friendliest of greetings.

### DEBUG command reference

You need to remember two commands from the outset. First, the Q command quits the DEBUG program and returns you to the DOS prompt. Second, the ? command is the help command, which instructs DEBUG to display a list of all DEBUG commands along with a brief description of each one (see Figure 11-3).

As Figure 11-3 shows, DEBUG has some 23 commands. They are explained in the following paragraphs.



```
-?
assemble     A [address]
compare      C range address
dump         D [range]
enter        E address [list]
fill         F range list
go           G [=address] [addresses]
hex          H value1 value2
input        I port
load         L [address] [drive] [firstsector] [number]
move         M range address
name         N [pathname] [arglist]
output       O port byte
proceed      P [=address] [number]
quit         Q
register     R [register]
search       S range list
trace        T [=address] [value]
unassemble   U [range]
write        W [address] [drive] [firstsector] [number]
allocate expanded memory        XA [#pages]
deallocate expanded memory      XD [handle]
map expanded memory pages       XM [Lpage] [Ppage] [handle]
display expanded memory status  XS
-
```

**Figure 11-3**: DEBUG's help command, ?, displays a brief description of every DEBUG command.

## The ? (help) command

The ? (help) command displays a summary of the DEBUG commands. The syntax is simply

```
?
```

## The A command

The A command is the assemble command. It assembles assembly-language statements and creates machine code. The syntax is as follows:

a [*address*]

This feature allows you to enter instructions using assembly-language notation such as

```
MOV AH,2A
INT 21
MOV AH,4C
INT 21
```

rather than directly entering the machine code bytes in the following format:

```
e 0100 B4 2A CD 21 B4 4C CD 21
```

You usually will use the A instruction to assemble a new program prior to writing it to disk. By default, DEBUG assembles from the beginning of the file or from the point where the last assemble operation stopped. Alternatively, you can specify the address of the range to be assembled. Here is an example:

```
-a0100:0108
```

## The C command

This is the compare command. It compares the data stored in two memory locations and displays the byte values side by side for each location. The syntax is

```
c range address
```

The *range* parameter is the starting and ending address of the first memory location, separated by a comma. The *address* parameter identifies the starting location of the second memory address. DEBUG displays each address where the byte values differ. For example, suppose that during the editing of the sample CONFIG.SYS file shown in Figure 11-2, the following command is executed:

```
-c 0100,010f 0119
```

DEBUG produces the following display:

```
11AA:010E  48  45  11AA:0127
11AA:010F  49  4D  11AA:0128
```

The compare command instructed DEBUG to compare the data at memory location 0100 through 010f with the data starting at memory location 0119. Both of these memory locations start with the string *DEVICE=C:\DOS\*, but the remaining two bytes in the comparison area are different. The first area held the bytes 48 and 49, which are the letters *HI* (the beginning of HIMEM.SYS), and the second area contained the bytes 45 and 4D, which are the letters *EM* (the beginning of EMM386.SYS).

## The D command

The D (dump) command instructs DEBUG to dump (that is, to display) the data stored at a specific memory location. The command syntax is

```
d [range]
```

The dump command is used to display the file contents in Figures 11-1 and 11-2. The optional *range* parameter specifies the starting and ending addresses or the starting address and length of the area of memory that you want to view. If you do not specify a range, DEBUG dumps 128 bytes of data, starting either at the byte following the data displayed with the last dump command or at the beginning of the file. By repeatedly pressing d, you can browse through a file 128 bytes at a time.

## The E command

The E (enter) command enters data into a specific memory location. The command syntax is

```
e address [list]
```

The *address* parameter is the starting address where the data will be entered, and the optional *list* parameter is the list of byte values identifying the data to be installed. Here is an example:

```
e 0100 B4 2A CD 21 B4 4C CD 21
```

If you don't specify any data, DEBUG shifts into Edit Mode and prompts you for input one byte at a time. DEBUG displays the existing value (followed by a period) and waits for you to enter the replacement value. I tend to use the E command for patching files but prefer to enter new programs using assembly-language mnemonics and use A to assemble them.

## The F command

The F command is the fill command. It fills an area of memory with a repeated byte or set of bytes. Use the following syntax:

```
f range list
```

The *range* variable specifies the beginning and ending addresses of the area to be filled, and *list* identifies the data that is to be filled into the area.

## The G command

The G (go) command runs the program. Its syntax is

```
g [=address] [breakpoints]
```

This feature really is designed as a debugging tool. It runs the program starting at a specific address and allows you to set up to ten breakpoints by specifying the individual breakpoint memory addresses. Just as I never use the tracking option on my VCR, I never use this command.

## The H command

H is the hex command. It performs addition and subtraction of two hexadecimal numbers and displays the results. The correct syntax is

```
h value1 value2
```

The following batch file uses the H command to create a simple but effective hex calculator:

```
@echo off
echo Hexadecimal Math
echo Plus  Minus
echo h %1 %2 >$$$$TEMP.SCR
echo q >> $$$$TEMP.SCR
debug < $$$$TEMP.SCR | find "-" /v
del $$$$TEMP.SCR
```

## The I command

The I (input) command reads and displays a single byte of data from a specific port. The syntax for this command is

`i` *port*

The *port* variable refers to the address of the port — for example, 03f8 for COM1.

## The L command

The L (load) command loads a file from disk into a specific address in memory. Use the following syntax:

`l [`*address*`]`

By default, the file loads at the address cs:100 (100 bytes into the code segment). Typically, you will load a file by specifying the filename on the command line when you execute DEBUG. When the file has an EXE extension, the load address is ignored. EXE files include a load location in the EXE header, and DEBUG uses this address. Rename an EXE file so that the file has a different extension if you want to edit the complete file, including the EXE header bytes.

DEBUG also enables you to load specific disk sectors with the L command. But because you can create havoc writing changed data back to disk, I recommend that you ignore this feature.

## The M command

The M (move) command copies a block of data from one location to another. The proper syntax is

`m` *range address*

If the source and target addresses overlap, the source data is overwritten.

### The N command

The N (name) command identifies the name of a file to be loaded or saved. Normally, you use this command when creating small COM file utilities. Use the N command to assign the filename prior to executing the W command to write the file to disk. The syntax is

```
n [drive:][path]filename.ext
```

### The O command

The O (output) command copies a specific byte to a port, using the following syntax:

```
o port datavalue
```

As with the I command, *port* is the address of the port — for example, 03f8 for COM1.

### The P command

The P (proceed) command instructs DEBUG to execute a program instruction. This feature is provided to support software debugging, and so you will rarely, if ever, use it. If you do, however, the syntax is as follows:

```
p [=address] [number]
```

### The Q command

The Q (quit) command is everybody's favorite. It quits DEBUG and returns to the DOS prompt. Changes are not automatically saved to disk. The syntax is simply

```
q
```

### The R command

The R (register) command displays a specific register and enables you to update that register. Its syntax is

```
r [register]
```

The following standard 80x86 registers are supported: AX, BX, CX, DX, SP, BP, SI, DI, DS, ES, SS, CS, IP, PC, and F (for flags). The AX through DX registers may also be referenced in byte-sized pieces using the high and low register components. To access AX in two sections, for example, specify AH and AL. When you execute R without a register name, the status of all registers is displayed. When you create a new COM file, the BX:CX registers must be updated with the file size before you call the W command; W looks at these registers to determine how many bytes to write.

## The S command

The S (search) command searches an area of memory for a specific byte, series of bytes, or string. The correct syntax is

```
s range list
```

*Range* represents the beginning and ending address of the area to be searched. *List* represents the byte (or series of bytes separated by spaces) you want to locate. Enclose strings in quotes. If you specify a single byte or a string, DEBUG displays all memory locations where a match is found. If you specify an array of bytes, the address of the first match is displayed. You can then use the D switch to display the data at that address.

## The T command

The T (trace) command executes (or single-steps) a single instruction and then displays the register contents. This is another of the program debugging options you are unlikely to use, but if you do, the proper syntax is

```
t [=address] number
```

## The U command

The U (unassemble) command displays a portion of the program using assembly-language mnemonics. This process is also known as *disassembling*. The correct command syntax is

```
u [range]
```

The U command is a very useful tool for inspecting small programs. You must, however, understand assembly language. If you do not specify a *range* value, 32 bytes of code are displayed, starting at the memory location immediately following the last unassembled code.

## The W command

The W (write) command writes a file to disk — in other words, saves the file. The command syntax is as follows:

```
w [address]
```

If you do not specify an address, the entire file is written. If you want to write a new file to disk, you must use the N switch to assign a name to the file and store the file's size in the BX:CX registers before executing W.

You also can use the W command to write data to specific disk sectors. Because this technique bypasses DOS and could destroy valuable data, however, I recommend that you ignore this feature.

### The XA command

The XA (allocate expanded memory) command allocates expanded memory pages and displays the handle number if successful. Unless you want to tool around with XMS for fun, you will find little use for the XA switch and the ones that follow. If you want to use them, however, the syntax is

```
xa [count]
```

### The XD command

The XD (deallocate expanded memory) command deallocates expanded memory and makes the pages available for other applications. The command syntax is

```
xd [handle]
```

### The XM command

The XM (map expanded memory pages) command maps a logical page to a physical page. Use the following syntax:

```
xm [logpage] [realpage] [handle]
```

### The XS command

The XS (expanded memory status) command shows the current status of expanded memory. Of all the XMS debug switches, this is the most interesting because it gives a snapshot of XMS usage. The correct syntax is simply

```
xs
```

NOTE

The DOS 6 on-line help system includes comprehensive help on the debug commands. To get help, enter the command `help debug`.

## How to enter DEBUG commands

DEBUG is not very particular about spaces in commands. In general, you do not need to separate a DEBUG command from the command parameters. If you prefer, however, you can separate the command and the parameters by a space or a comma for clarity. For example, the following three statements all have the same meaning:

```
-nutil.com
-n util.com
-n,util.com
```

TECHNICAL STUFF

You must specify memory ranges or addresses when using many of DEBUG's commands. Enter addresses in the *segment:offset* format and specify them in hexadecimal. (Refer to Chapter 5 for a discussion of segments, offsets, and hexadecimal numbers.) All DOS programs have two special segment addresses: the code segment and the data segment. These segments identify the starting address of a program's code and data. DEBUG supports the segment aliases CS and DS, respectively. For example, the memory address that is 200 bytes into the code segment is CS:0200.

If you specify a single address (without a colon), DEBUG assumes that it is the memory offset and applies a default segment. The default segment is the code segment when you use commands A, G, L, T, U, and W; the default is the data segment for all other commands.

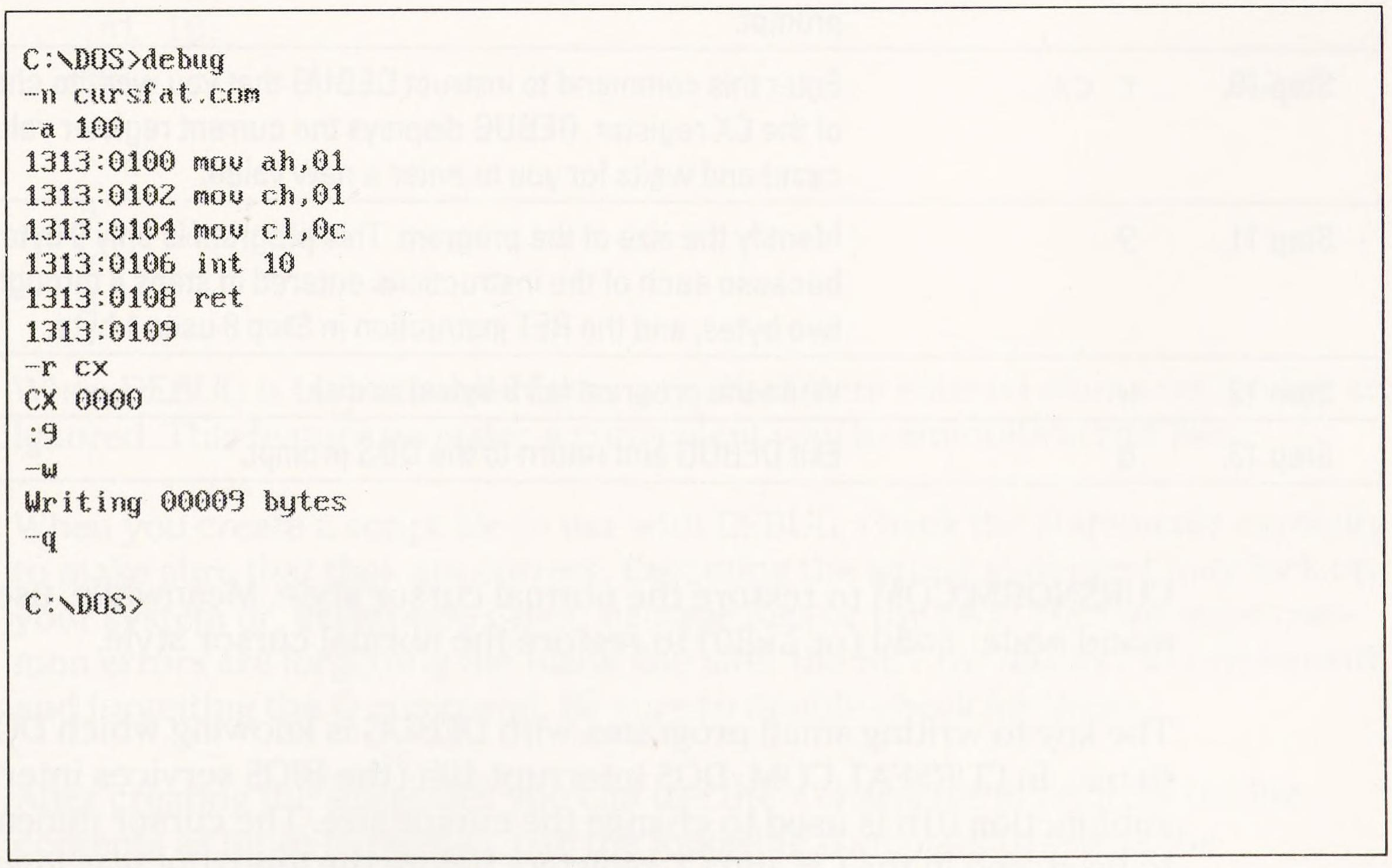
## Taking a test drive

Enough theory, already. It's time to put DEBUG to use. As is the case with most programming languages, the best way to learn about DEBUG is to use it.

By default, the DOS cursor is a thin blinking line located at the bottom of the text. You can use DEBUG to create a small program that makes the cursor a large block. Figure 11-4 shows the DEBUG session used to create the program CURSFAT.COM.

To create the program, follow the steps listed in Table 11-1.

Voilà! you have created the program CURSFAT.COM. Execute it, and the cursor changes to a large block. In the next section, you create the program



```
C:\DOS>debug
-n cursfat.com
-a 100
1313:0100 mov ah,01
1313:0102 mov ch,01
1313:0104 mov cl,0c
1313:0106 int 10
1313:0108 ret
1313:0109
-r cx
CX 0000
:9
-w
Writing 00009 bytes
-q

C:\DOS>
```

**Figure 11-4:** Using DEBUG to create CURSFAT.COM.

### Table 11-1 Creating the CURSFAT.COM Program

| *Step* | *Input* | *Description* |
|---|---|---|
| Step 1. | `debug` | Enter the DEBUG command at the DOS prompt to start a DEBUG session. |
| Step 2. | `n cursfat.com` | Assign the filename CURSFAT.COM. |
| Step 3. | `a 100` | Instruct DEBUG to start assembling code beginning at address CS:0100, which is the beginning address for COM programs. As soon as you enter this command, DEBUG shifts into Assemble Mode and waits for assembly-language mnemonics to be entered. |
| Step 4. | `mov ah,01` | To change the cursor, first set the value of AH to 01. |
| Step 5. | `mov ch,01` | This step sets the CH register to the starting cursor line — that is, the scan line position of the top of the cursor. |
| Step 6. | `mov cl,0C` | This step sets the CL register to the ending cursor scan line. ***Note:*** A color system uses 14 scan lines per character block, but a monochrome system uses only 8 scan lines. The scan lines are numbered 0 to C (13 decimal) and 0 to 7, respectively. Enter `mov cl,07` if you are writing this program for a monochrome system. |
| Step 7. | `int 10` | Invoke the DOS interrupt 10h. When you do so, DOS inspects the AH register to determine which task to perform. In this case, it finds the value 01, which is the code to change the cursor size. |
| Step 8. | `ret` | This step passes control back to DOS. |
| Step 9. | Press Enter. | Press Enter to stop the assemble session and return to the DEBUG prompt. |
| Step 10. | `r cx` | Enter this command to instruct DEBUG that you want to change the value of the CX register. DEBUG displays the current register value (0000 in this case) and waits for you to enter a new value. |
| Step 11. | 9 | Identify the size of the program. This program is only 9 bytes long, because each of the instructions entered in steps 4 through 7 consumes two bytes, and the RET instruction in Step 8 uses 1 byte. |
| Step 12. | w | Write the program (all 9 bytes) to disk. |
| Step 13. | q | Exit DEBUG and return to the DOS prompt. |

CURSNORM.COM to restore the normal cursor style. Meanwhile, use the command `mode co80` (or `bw80`) to restore the normal cursor style.

The key to writing small programs with DEBUG is knowing which DOS features to use. In CURSFAT.COM, DOS interrupt 10h (the BIOS services interrupt), subfunction 01h is used to change the cursor size. The cursor dimensions had to be stored in the CH and CL registers before the interrupt was invoked. Unless you are an experienced DOS programmer, you will not know details such as these. If you are interested in writing your own utilities, you need a DOS programming reference guide that documents all of the DOS interrupts. My favorite reference book is *PC Interrupts,* written by Ralph Kyle and Jim Brown.

## Creating ASCII script files

Writing DEBUG programs interactively (as you did for CURSFAT.COM) is practical only for very small programs. When you create larger programs, entering an incorrect instruction can be extremely frustrating. Although you can make corrections with the E switch, it is far from smooth sailing, and often the process becomes so confusing that you give up, start over, and reenter all the instructions.

Another option is to use EDIT or any editor to create an ASCII file and add one command per line. After you create and verify the commands, you can redirect the statements to DEBUG and create the program. ASCII files containing DEBUG commands often are referred to as *script files.*

The following script file, called CURSNORM.SCR, contains all the instructions to create the program CURSNORM.COM, which sets the cursor back to the normal thin style:

```
n cursnorm.com
a 100
mov ah,01
mov ch,0c; use ch,06 on monochrome
mov cl,0d; use cl,07 on monochrome
int 10
ret

r cx
9
w
q
```

NOTE

When DEBUG is in Assemble Mode, any characters entered after a semicolon are ignored. This feature provides a convenient way to annotate script files.

CAUTION

When you create a script file to use with DEBUG, check the statements carefully to make sure that they are correct. Executing the wrong statement may lock up your system or, in extreme cases, damage files or the FAT. The two most common errors are forgetting the blank line after the RET (or last INT 21) statement and forgetting the Q command. Be sure to double-check for these.

After creating the script file, you can use the < character to redirect the file contents as input to DEBUG. Use the following command to create the CURSNORM.COM file:

```
debug < cursnorm.scr
```

## Patching Programs with DEBUG

One common use of DEBUG is to edit or *patch* an existing program. Using DEBUG alone, you can perform tasks such as fixing a program bug, changing error messages, or modifying program colors. Before you change a program, make a backup copy of the file just in case you don't get the expected results.

To modify a program, you first must locate the specific data you want to change. To change an ASCII file with a text editor, you would probably use PgUp and PgDn to locate the text or, better still, use a search command. You can use the same approach for a program file, but the equivalent DEBUG commands are D (dump) and S (search).

After you identify the specific memory location, use the E (edit) command to modify the data and the W (write) command to save the data to disk. Because the file was loaded by DEBUG, you can use the W command without first storing the file size in the BX:CX registers. DEBUG assumes that the file is the same size as it was when it was loaded.

For an illustration of the approach, you can make some changes to two popular DOS commands.

### Changing the CHOICE command

GEM

By default, the CHOICE batch command prompts you to press either Y or N. With DEBUG, you can easily change these default keys. You can assign 1 and 2 as the default keys, for example.

To do this, copy the CHOICE.COM program (in the DOS directory) to CHOICE12.COM. Then load the new program into DEBUG by entering the following command:

```
debug choice12.com
```

At DEBUG's - prompt, issue the D command to dump the first 128 bytes. The following output is displayed:

```
0E4D:0100  E9 32 04 59 4E 00 00 00-00 00 00 00 00 00 00 00.2.YN ...........
0E4D:0110  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
0E4D:0120  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
0E4D:0130  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
0E4D:0140  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
0E4D:0150  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
0E4D:0160  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
0E4D:0170  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
```

The memory segment address (displayed in the first column) may differ on your system.

Notice the characters Y and N on the first line; these are actually the defaults used when you execute CHOICE without the /C switch. To change the values to *1* and *2,* enter the following command:

```
-e 0103 31 32
```

This command instructs DEBUG to edit data starting at offset 0103 and change the first two bytes to values 31 and 32, which are the hexadecimal ASCII values for the characters 1 and 2. To visually verify the changes, enter the following command:

```
d 0100
```

The final step is to save the changes with the W command and then press q to exit DEBUG.

In just a few simple steps, you change the default values of CHOICE from Y and N to 1 and 2.

To be thorough, however, you need to modify the help text so that it says *12* rather than *YN.* Launch DEBUG with the following command:

```
debug choice12.com
```

The S (search) command can search through the file looking for any additional YN references. At the DEBUG prompt, enter the following command and be sure to specify uppercase YN (the SEARCH command is case sensitive):

```
-s 0100 FFFF “YN”
```

This command instructs DEBUG to search the entire code segment for all occurrences of the “YN” string. DEBUG responds by displaying all addresses where the text is found. DEBUG should respond with one address similar to the following:

```
11CC:0228
```

The code segment (11CC in this example) may be different on your system.

Enter the D (dump) command to display the data in this general area:

```
-d 0220
```

DEBUG displays the data, and you see that the “YN” entry is indeed part of the help text. Enter the following EDIT command to change the help text to “12”:

```
-e 0228 31 32
```

Finally, use the W command to save your changes and the Q command to quit DEBUG. You can view your handiwork by entering the following command:

```
choice12 /?
```

You may have noticed that there is a lot of empty space at the beginning of the CHOICE.COM program. If you are wondering whether these 00s can be replaced with additional default keys, the answer is yes. You can even add special keys that are not directly supported by CHOICE, such as the spacebar and Enter, on the command line. In other words, you can use DEBUG to alter CHOICE so that it accepts the input of the spacebar or Enter key.

The following script file, CHOICEAZ.SCR, changes the default CHOICE keys to the spacebar, Enter, 0–9, and A–Z.

```
n choiceaz.com
l
d
e 0103 20 d 30 31 32 33 34 35 36 37 38 39
e 0110 40 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4D 4F 50
e 0120 51 52 53 54 55 56 57 58 59 5A
d 0100
w
q
```

Copy CHOICE.COM to CHOICEAZ.COM and then run the following command to redirect the script to DEBUG and update CHOICEAZ.COM:

```
debug < choiceaz.scr
```

The CHOICEAZ command is an ideal option when you have batch files and you want the system to pause for the user to press any key but then to proceed after a specified timeout period if no key is pressed. For example, you can use the following statement to pause a batch file for 10 seconds or until the user presses a key (including the spacebar or Enter), whichever occurs first:

```
choiceaz /n /T:A,10
```

## Patching the default EDIT filemask

I mainly use EDIT to build and modify batch files. One thing that irritates me about the editor is the default *.TXT filemask used in the File Open dialog box. However, with a little help from DEBUG, you can easily change the prompt to *.BAT.

EDIT.EXE is not actually the program that provides the full screen-editing functions. When you enter the EDIT command, EDIT in turn calls the QBASIC.EXE program with the switch /EDITOR. It is QBASIC.EXE that provides all the editing tools, and lurking deep inside QBASIC.EXE is the default filemask of *.TXT.

DEBUG does not support the editing of files that have an EXE extension; DEBUG loads an EXE in a special way, so the data loaded is not an exact facsimile of the file. The workaround is to copy QBASIC.EXE and give the copy a name with a different extension, say QBASIC.DBG.

After you do that, start the DEBUG session by entering the following command:

```
debug qbasic.dbg
```

In the example discussed in the preceding section, you used the DEBUG search command to locate the "YN" string. You might think that a similar approach would work in this instance. To test this theory, enter the following command to search for *.TXT (use uppercase):

```
-s 0100 FFFF "*.TXT"
```

DEBUG does not find any occurrences of "*.TXT", and so no memory addresses are displayed. Why? The search command only searched the first 64K of memory, and QBASIC.EXE is almost 200K in size. DEBUG did not find the filemask because it wasn't stored in the first 64K. To instruct DEBUG to search the next 64K requires a little hexadecimal arithmetic.

The first step is to determine the specific address of the code segment. Enter the D command to dump the first 128 bytes of QBASIC.EXE. Output similar to the following is displayed:

```
11CC:0100  B8 EE 44 BA 69 2F 05 DC-11 3B 06 02 00 72 1B B4   ..D.i/...;...r..
11CC:0110  09 BA 18 01 CD 21 CD 20-4E 6F 74 20 65 6E 6F 75   .....!. Not enou
11CC:0120  67 68 20 6D 65 6D 6F 72-79 24 2D 20 00 8E D0 2D   gh memory$- ...-
11CC:0130  25 00 90 8E C0 50 B9 23-01 33 FF 57 BE 44 01 FC   %....P.#.3.W.D..
11CC:0140  F3 A5 CB 90 FD 8C DB 53-81 C3 39 00 03 DA 8C CD   .......S..9.....
11CC:0150  8B C2 80 E4 0F B1 04 8B-F2 D3 E6 D3 EA FE C6 8B   ................
11CC:0160  CE D1 E9 4E 4E 8B FE 2B-E8 2B D8 8E C5 8E DB F3   ...NN..+.+......
11CC:0170  A5 B8 00 10 B5 80 FE CE-75 ED FC 8E DD 07 06 BF   ........u.......
```

CAUTION

The remaining steps involve working with segment addresses. The specific address may be different on your system, so be sure to use the address displayed on your system rather than the example addresses used in this text.

You can use the DEBUG hex calculator to find a segment address for the next 64K. Add 0FFFh to the code segment — each segment address is multiplied by 16 (decimal) to get a linear address. Use the H command, specifying the segment address and 0FFF. Here is an example:

```
-h 11CC 0FFF
```

DEBUG responds by displaying two values. The first value is the sum of the two numbers, and the second is the difference between the two numbers. Here is an example:

```
21CB 01CD
```

The first value represents the beginning segment of the next 64K of QBASIC.DBG (or whatever name you used for the copy of QBASIC.EXE). Use this value in another search command as follows:

```
-s 21CB:0000 FFFF “*.TXT”
```

Once again, no match is found. Use the H command to compute the beginning segment for the next 64K. Here is an example:

```
-h 21CB 0FFF
```

In my case, the value returned was 31CA. Try the search one more time, searching the third 64K block of QBASIC.EXE. Here is an example:

```
-s 31CA:0000 FFFF “*.TXT”
```

Bingo! DEBUG responds with a single address indicating that it found the text:

```
31CA:DD1A
```

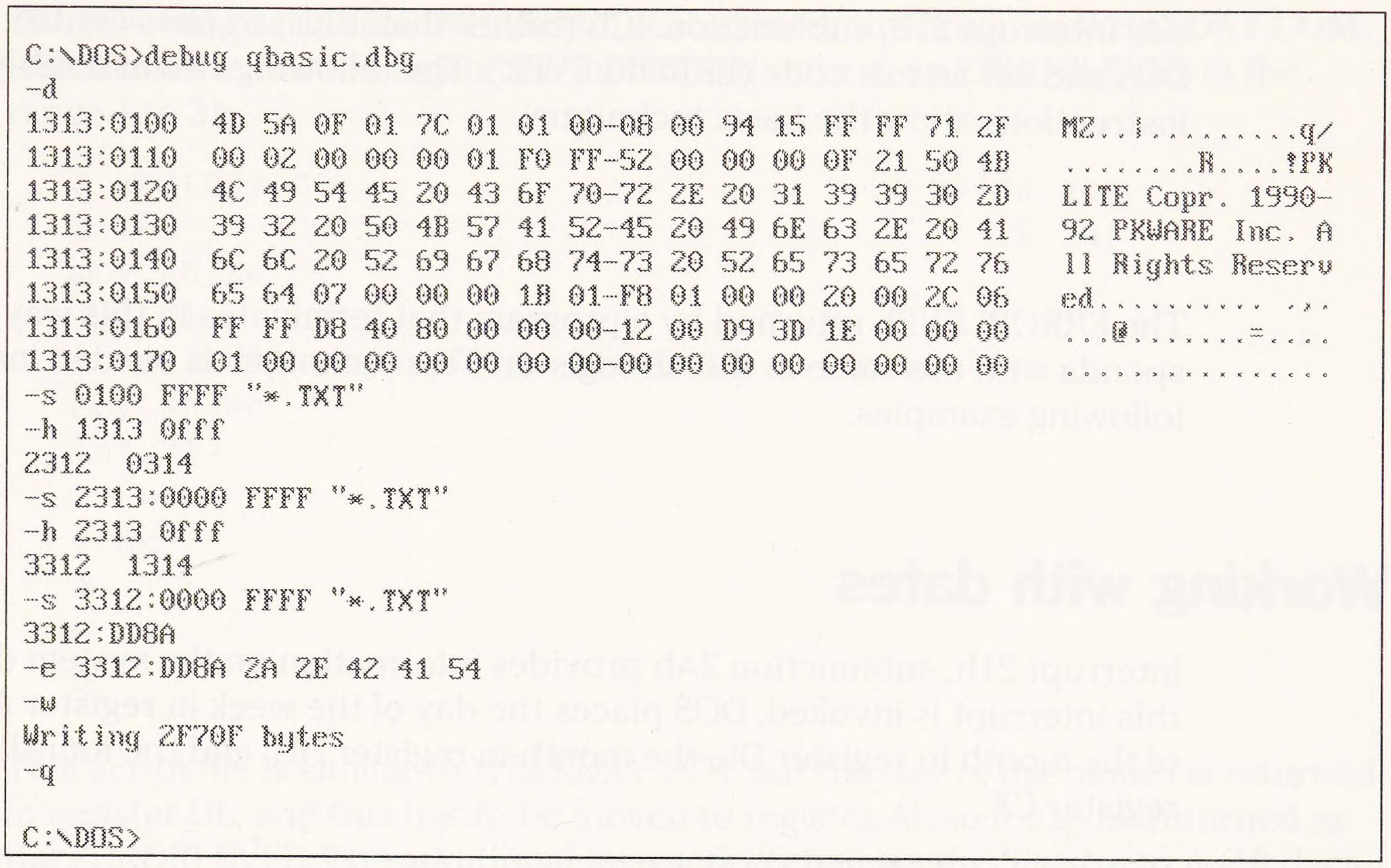
You can view this area using the dump command as follows:

```
-d 31CA:DD10
```

The biggest challenge was finding the “*.TXT” string. Changing the string is easy using the EDIT command. To make things simple, use the specific address displayed by the search command (for example, 31CA:DD1A) and change the first five characters to *.BAT as follows:

```
-e 31CA:DD1A 2A 2E 42 41 54
```

Save the file and quit DEBUG by entering the W and Q commands.



```
C:\DOS>debug qbasic.dbg
-d
1313:0100  4D 5A 0F 01 7C 01 01 00-08 00 94 15 FF FF 71 2F   MZ..|.........q/
1313:0110  00 02 00 00 00 01 F0 FF-52 00 00 00 0F 21 50 4B   ........R....!PK
1313:0120  4C 49 54 45 20 43 6F 70-72 2E 20 31 39 39 30 2D   LITE Copr. 1990-
1313:0130  39 32 20 50 4B 57 41 52-45 20 49 6E 63 2E 20 41   92 PKWARE Inc. A
1313:0140  6C 6C 20 52 69 67 68 74-73 20 52 65 73 65 72 76   ll Rights Reserv
1313:0150  65 64 07 00 00 00 1B 01-F8 01 00 00 20 00 2C 06   ed.......... .,.
1313:0160  FF FF D8 40 80 00 00 00-12 00 D9 3D 1E 00 00 00   ...@.......=....
1313:0170  01 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-s 0100 FFFF "*.TXT"
-h 1313 0fff
2312  0314
-s 2313:0000 FFFF "*.TXT"
-h 2313 0fff
3312  1314
-s 3312:0000 FFFF "*.TXT"
3312:DD8A
-e 3312:DD8A 2A 2E 42 41 54
-w
Writing 2F70F bytes
-q

C:\DOS>
```

Figure 11-5: Changing the QBASIC.EXE default wildcard from *.TXT to *.BAT.

Before renaming QBASIC.DBG to QBASIC.EXE, rename QBASIC.EXE to QBASIC.BAK (just in case you didn't make the correct modifications). After you rename the files, EDIT uses the default wildcard of *.BAT. Figure 11-5 shows a complete session summarizing how to change the filemask.

## Creating Useful Utilities with DEBUG

DEBUG is a great tool for creating compact, high-performance utilities. In this section, you learn how to create programs for returning the system date and for interacting with the keyboard. You also learn how to create disk drive utilities.

### Returning to DOS and setting an ERRORLEVEL

Many utilities created with DEBUG are designed to be used in batch files. Because the IF statement can test a program's ERRORLEVEL, many of the utilities set an ERRORLEVEL to pass information to the batch file.

Use interrupt 21h, subfunction 4Ch (rather than RET) to pass control back to DOS and set an exit code (ERRORLEVEL). The following two assembly-language instructions show the basic technique:

```
mov ah,4c
int 21
```

The ERRORLEVEL returned by a program that terminates in this way corresponds with the value in the AL register. This technique is used in many of the following examples.

## Working with dates

Interrupt 21h, subfunction 2Ah provides information on the system date. When this interrupt is invoked, DOS places the day of the week in register AL, the day of the month in register DL, the month in register DH, and the four-digit year in register CX.

Armed with this knowledge, you can easily use DEBUG to create some very useful date routines.

The following script file, WEEKDAY.SCR, returns the day of the week as an ERRORLEVEL:

```
n weekday.com
a 100
mov ah,2a
int 21        ;day of week is now in AL
mov ah,4c
int 21

r cx
8
w
q
```

To create the WEEKDAY.COM program, execute the following command:

```
debug < weekday.scr
```

WEEKDAY.COM sets an ERRORLEVEL to indicate the day of the week. Sunday is represented by an ERRORLEVEL of 0, Monday by an ERRORLEVEL of 1, and so on through Saturday, which has an ERRORLEVEL of 6.

Refer to Chapter 10 for an example of how to use WEEKDAY.COM in a batch file.

The following script file, WHATDAY.SCR, creates the program WHATDAY.COM, which determines the day of the month and returns an ERRORLEVEL in the range 1 to 31:

```
n whatday.com
a 100
mov ah,2a
int 21
mov al,dl   ;return day of month as error level
mov ah,4c
int 21

r cx
A
w
q
```

This script file is similar to WEEKDAY.SCR, but the day of the month is returned in register DL, and this has to be moved to register AL so it can be returned as the ERRORLEVEL. The additional instruction changes the file size to A (10 decimal) bytes.

The following script file, WHATMON.SCR, creates the program WHATMON.COM, which determines the month and returns an ERRORLEVEL in the range 1 to 12:

```
n whatmon.com
a 100
mov ah,2a
int 21
mov al,dh   ;return month as error level
mov ah,4c
int 21

r cx
A
w
q
```

This script is almost identical to WHATDAY.SCR. The only difference is that the month value is returned in register DH.

The final script in the quartet of date utilities is WHATYEAR.SCR, which returns the year as an ERRORLEVEL. DOS returns the four-digit year in the CX register. However, ERRORLEVELs are only supported in the range 0 to 255. By moving the year to the AX register and subtracting 1980 from it (7CB in hexadecimal), the ERRORLEVEL is set to the year since 1980. For example, 1994 is returned as ERRORLEVEL 14. The WHATYEAR.SCR file is as follows:

```
n whatyear.com
a 100
mov ah,2a
int 21
mov ax,cx    ; move year (format YYYY) to AX
sub ax,07bc  ; subtract 1980
mov ah,4c    ; year since 1980 now in AL
int 21

r cx
D
w
q
```

# Writing keyboard management utilities

The DOS batch file language provides very few ways of obtaining user input or checking the keyboard status. The utilities in this section fill this void in the language.

## Checking and changing the state of special keys

Tucked away in BIOS, at the address 0040:0017, is a very useful byte of information. This byte indicates the status of the special keys Left Shift, Right Shift, Ctrl, Alt, Scroll Lock, Num Lock, Caps Lock, and Ins. Each of the eight bits is set to *on* or *off* to indicate whether each of the special keys is being held down or, in the case of lockable keys, is set on. By returning this byte as an ERRORLEVEL, a small utility can inform a batch file of the status of these special keys.

You can use the following script file, KEYSTATE.SCR, to create KEYSTATE.COM:

```
n keystate.com
a 100
mov ax,0040
mov ds,ax
mov al,[17] ; returns the keyboard status bit as errorlevel
mov ah,4c
int 21

r cx
C
w
q
```

When [ ] brackets are used in a MOV statement, DEBUG interprets the value to be a memory location and moves the data at the specified memory location into a register. In this example, the 17th byte in the data segment (which points to segment 0400) is moved into the AL register.

The ERRORLEVELs set by each key are listed in Table 11-2.

KEYSTATE provides a great way to add shortcuts to batch files. For example, consider the following batch file, BYPASS.BAT:

```
1  @echo off
2  keystate
3  if errorlevel 16 if not errorlevel 17 goto jump
4  choice "Do you really want to delete all *.BAK files "
5  if errorlevel 2 goto quit
6  :JUMP
7  del *.BAK
8 :QUIT
```

Line 5 checks to see whether the Scroll Lock key is on (because KEYSTATE sets an ERRORLEVEL of 16 if Scroll Lock is on) and, if it is, skips the confirmation prompt and gets on with the file deletion.

If more than one special key is active or held down, the ERRORLEVEL returned by KEYSTATE is the sum of the individual key ERRORLEVELs. For example, if Caps Lock is on and the Alt key is held down, the ERRORLEVEL is set to 72.

**Table 11-2 ERRORLEVELs Set by KEYSTATE.COM**

| *ERRORLEVEL* | *Key Status* |
|---|---|
| 0 | No keys are active. |
| 1 | Right Shift is held down. |
| 2 | Left Shift is held down. |
| 4 | Ctrl is held down. |
| 8 | Alt is held down. |
| 16 | Scroll Lock is active. |
| 32 | Num Lock is active. |
| 64 | Caps Lock is active. |
| 128 | Ins is held down. |

By changing the value of the byte at address 0040:0017, you can alter the state of the Caps Lock, Num Lock, and Scroll Lock keys. The following script file creates CAPSOFF.COM, which turns off the Caps Lock:

```
n capsoff.com
a 100
mov ax,40
mov ds,ax
mov al,[17]
and al,BF     ; set bit 1 off
mov [17],al
ret

r cx
E
w
q
```

The following script file creates CAPSON.COM, for turning Caps Lock on:

```
n capson.com
a 100
mov ax,40
mov ds,ax
mov al,[17]
or al,40   ; Set bit 1 on
mov [17],al
ret

r cx
E
w
q
```

The AND and OR statements in these two script files turn the Caps Lock bit off and on, respectively. You can use similar techniques to create utilities to set Num Lock and Scroll Lock on or off. The following statements create on and off utilities for Num Lock and Scroll Lock:

| | |
|---|---|
| `and al,df` | Turns Num Lock off |
| `or al,20` | Turns Num Lock on |
| `and al,ef` | Turns Scroll Lock off |
| `or al,10` | Turns Scroll Lock on |

## Rebooting the PC

You can use the following script file, REBOOT.SCR, to create the utility REBOOT.COM:

```
n reboot.com
a 100
mov ah,0d          ; flush disk cache
int 21             ; Call Interrupt
mov ax,0040
mov ds,ax
mov ax,1234
mov [72],ax
jmp FFFF:0000

r cx
14
w
q
```

When you execute REBOOT.COM, your system reboots.

This small program first calls sets AH equal to 0 and calls interrupt 21h, to ensure that any data in a write-delayed disk cache is saved to disk. See Chapter 4 for a discussion of write-delayed caching. Next, the value 1234h is moved to the address 0040:0072 and then executes the code at address FFFF:0000, which is the power-on self test (POST) BIOS code used at system boot time. When the POST instructions are executed, the value stored at 0040:0072 is inspected, and if it has a value of 1234h, a warm reboot is performed. If that memory location has any other address, a cold boot is performed. You can modify REBOOT.SCR to perform a cold boot by replacing the MOV AX,1234 statement with MOV AX,ABCD.

**CAUTION** When REBOOT.COM is executed in a Windows DOS session, Windows does not trap the reboot attempt as it does when you press Ctrl-Alt-Del. The system simply reboots.

## Waiting for a function key

Call me old fashioned, but I like function keys. Unfortunately, CHOICE does not provide function key support. The following script, GETFUNCY.SCR, creates the program GETFUNCY.COM:

```
n getfuncy.com
a 100
jmp 0108      ; skip the beep
mov dl,7      ; the beep meister
mov ah,2       ; sub-function 2
int 21         ; beep
mov ah,00      ; sub-function 0
int 16         ; wait for keyboard input
cmp al,00      ; is it an extended key
ja 0102        ; not extended—go get another key
cmp ah,3b      ; compare with 59 decimal (F1)
jl 0102        ; too small—go get another key
cmp ah,44      ; compare with 68 decimal (F10)
ja 0102        ; too large—go get another key
sub ah,3a      ; subtract 58 decimal from key value
mov al,ah      ; set key value as errorlevel
mov ah,4c
int 21

r cx
23
w
q
```

GETFUNCY waits for the user to press any one of the function keys F1 through F10. If the user presses a different key, the program beeps and waits for the user to press another key. An ERRORLEVEL is returned to indicate which function key the user selected — from ERRORLEVEL 1 for F1 through ERRORLEVEL 10 for F10.

The following batch file, TESTFUNC.BAT, shows GETFUNCY in action:

```
@echo off
echo Press a Function Key
getfuncy
if errorlevel 10 echo You pressed F10
if errorlevel 9 if not errorlevel 10 echo You pressed F9
if errorlevel 8 if not errorlevel 9 echo You pressed F8
if errorlevel 7 if not errorlevel 8 echo You pressed F7
if errorlevel 6 if not errorlevel 7 echo You pressed F6
if errorlevel 5 if not errorlevel 6 echo You pressed F5
if errorlevel 4 if not errorlevel 5 echo You pressed F4
if errorlevel 3 if not errorlevel 4 echo You pressed F3
if errorlevel 2 if not errorlevel 3 echo You pressed F2
if errorlevel 1 if not errorlevel 2 echo You pressed F1
```

## Creating disk and drive utilities

The following paragraphs examine two utilities to help with drive management.

### Abort, Retry, Fail? Not!

There are few prompts uglier than an `Abort, Retry, Fail?` message emanating from a batch file when it is trying to access an empty drive.

The following script, DRIVEAOK.SCR, creates the program DRIVEAOK.COM, which tests whether drive A has a disk installed:

```
n driveaok.com
a 100
mov ax,2524     ; Change interrupt handler for Int. 24
mov dx,0113     ; New handler at address 0113 (see below)
int 21          ; Call Interrupt
mov ah,36       ; Get free space on drive
mov dl,01       ; Drive A is test drive
int 21          ; Call interrupt
mov ax,4c00     ; Got this far drive must be OK—set 0 errorlevel
int 21          ; Call it a day
mov ax,4c01     ; Critical error—drive A not ready—set 1 errorlevel
int 21          ; Call it a day

r cx
18
w
q
```

There is more to this compact script than meets the eye. When an empty drive is accessed, the DOS critical error handler, interrupt 24h, is invoked. The first task is to change the critical error handler to some custom code and bypass the usual `Abort, Retry, Fail?` message. Interrupt 21h, subfunction 25 instructs DOS to change the interrupt handler for the interrupt specified in register AL. The new interrupt handler is identified in register DX. In this case, DX points to byte 113 of the data segment, which is simply the code to set an ERRORLEVEL of 1 and terminate the program.

The next step is to try to access the drive and thereby provoke a critical error if the drive is not ready. Interrupt 21h, subfunction 36h gets the free space on the drive specified in register DL. If the free space information is returned, the program terminates with a 0 ERRORLEVEL. Otherwise, a critical error occurs, and the custom error handler is invoked.

The program sets an ERRORLEVEL of 1 if the drive is open and 0 if it contains a disk. The following batch fragment file shows how to use DRIVEAOK in a batch file and avoid the standard DOS error message:

```
@echo off
:START
driveaok
if errorlevel 1 goto message
rem Insert A related commands below
echo A is OK
goto quit
:MESSAGE
echo Insert a formatted disk in drive A
pause
goto start
:QUIT
```

You can use the following script, DRIVEBOK.SCR, to create the program DRIVEBOK.COM, which tests whether drive B has a disk installed:

```
n drivebok.com
a 100
mov ax,2524     ; Change interrupt handler for Int. 24
mov dx,0113     ; New handler at address 0113 (see below)
int 21          ; Call Interrupt
mov ah,36       ; Get free space on drive
mov dl,02       ; Drive B is test drive
int 21          ; Call interrupt
mov ax,4c00     ; Got this far drive must be ok—set 0 errorlevel
int 21          ; Call it a day
mov ax,4c01     ; Critical error—drive B not ready—set 1 errorlevel
int 21          ; Call it a day

r cx
18
w
q
```

## A script to determine the boot drive

If you are writing a batch file that copies, modifies, or searches the start-up files, you need to determine the boot drive. Don't just assume that the boot drive is C.

The following script, BOOTDRV.SCR, uses interrupt 21h, function 33h to determine the boot drive and creates the file BOOTDRV.COM:

```
n bootdrv.com
a 100
mov ax,3305      ; Get boot drive
int 21           ; Call Interrupt
mov al,dl        ; Move boot drive to AL
mov ah,4c        ; Exit with errorlevel
int 21           ; Call it a day

r cx
B
w
q
```

Register DL is updated with the boot drive, which is then returned as an ERRORLEVEL. Drive A is ERRORLEVEL 1, drive B is ERRORLEVEL 2, and so on.

The following batch file, TESTBOOT.BAT, shows BOOTDRV.COM in action:

```
@echo off
bootdrv
if errorlevel 4 if not errorlevel 5 echo Booted from drive D
if errorlevel 3 if not errorlevel 4 echo Booted from drive C
if errorlevel 2 if not errorlevel 3 echo Booted from drive B
if errorlevel 1 if not errorlevel 2 echo Booted from drive A
```

## Summary

This chapter showed how you can use DEBUG for many purposes other than tracking bugs in programs. You examined the following applications for DEBUG:

- You can browse program and ASCII files with the D (dump) and S (search) commands. Unlike EDIT, DEBUG displays every byte in the file, including the carriage return and line feed characters.
- You can patch programs such as QBASIC.EXE and CHOICE.COM to change their behavior and properties by using the E (edit) command. EXE files, however, must first be renamed with a different extension.
- DEBUG is ideal for creating small and compact utilities to extend and enhance batch files.
- Programs can be created interactively or by redirecting script files to DEBUG.
- The A (assemble) command enables you to enter commands using assembly-language mnemonics.

Part IV of this book contains a complete and thorough command reference for all the DOS 6 programs and device drivers.

# Part IV:

# The Complete DOS 6 Command Reference

6
DOS
Secrets

# The Complete DOS 6 Command Reference

## Introduction

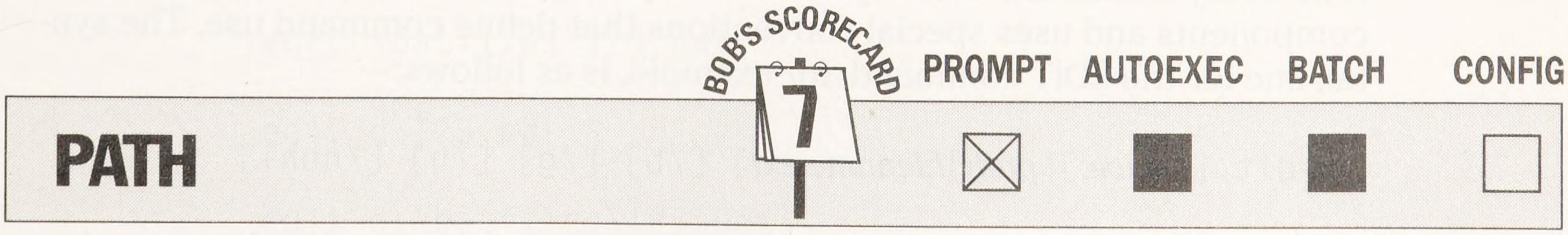
This section is an at-a-glance summary of every DOS command available in DOS 6, including all the popular DOS commands, such as XCOPY and MD, as well as device drivers, such as ANSI.SYS. The section even includes programming commands, such as GOTO, used in batch files. In short, if DOS has a command or program — and DOS 6 contains more commands than ever — you'll find it listed in this reference.

With so many commands to choose from, the task of sorting the more important commands from the least used commands can be difficult. To help you separate the wheat from the chaff, I have *rated* each command on a scale from 1 to 10. The best and most relevant commands receive a score of 10. At the other end of the scale, ratings in the range from 1 to 3 indicate poor or obscure commands.

A *command banner* introduces every command. Each command banner includes the name of the command, the command's rating on *Bob's Scorecard,* and a set of four check boxes that identifies when you use the command. Each check box can appear in one of the following three ways:

- A filled-in, black box indicates the primary use of the command.
- An X in the box indicates less frequent use of the command.
- An open or empty box signifies that the command is not used in that category.

The command banner for the PATH command looks like this:



The check boxes for PATH indicate that it is used primarily in the AUTOEXEC.BAT file, primarily in batch files, occasionally at the DOS prompt, but never in the CONFIG.SYS file.

I give the PATH command a rating of 7 because every respectable DOS user needs to use the PATH command, but it can be improved. The path length has a 127-character limit, for example, and no direct support exists for path sets — batch files must be used to switch between paths. PATH is a good command, but not a great one.

To further assist you in using this reference, all commands appear in alphabetical order, regardless of the command's category. The DOS command HELP precedes the device driver HIMEM.SYS, for example, which is followed by IF, a batch file command.

## Command Use Conventions

It is often more difficult to describe a command than it is to use it! This statement is especially true when you can use a command in many different ways and when it has a variety of *options* and *switches.*

Consider the EDIT command you use to edit a text file. In its simplest form, you just enter the command `edit`, the editor runs, and away you go. Alternatively, you can specify the name of the file you want to edit when you enter the command, as shown in the following example:

```
edit c:\autoexec.bat
```

The editor automatically loads the file, ready for editing. Because you can include a drive and a directory path when you enter a filename, you can load files not located in the current directory.

In addition, the EDIT command supports a variety of switches that control some of the editor properties. You can use the /B switch, for example, to force the editor to appear on-screen in black and white rather than color. As with many DOS commands, EDIT clearly has more capability than first meets the eye.

To help you understand all the options and switches available with the DOS commands, a "Usage" section that defines the complete command syntax follows every command "Description." The syntax line includes all optional components and uses special conventions that define command use. The syntax line for the EDIT command, for example, is as follows:

```
edit [[drive:][path]filename.ext] [/b] [/g] [/h] [/nohi]
```

This description can seem daunting at first. With an explanation, however, it provides a convenient and concise way to define each command and its options. The following conventions used in *DOS 6 SECRETS* are similar to the ones used in the DOS 6 on-line help.

TIP

You can always display an explanation of the syntax and usage of each command by using the DOS HELP command or by entering a command followed by the /? switch.

## Mandatory elements

Any part of a command that is *mandatory,* such as the command name, does *not* appear in square brackets ([]). For example, the only part of the syntax line for the EDIT command that appears without square brackets is `edit`. The syntax line for the DEL command, however, always contains at least two mandatory elements not enclosed in square brackets: the command name `del` and the name of the file to be deleted.

## Optional elements

Any optional components of a command *are enclosed* in square brackets. The /B switch in the EDIT command, for example, is displayed in the use syntax as `[/b]` to indicate that it is optional (syntax lines appear in lower case).

Most DOS commands support the /? switch to display command help. To avoid unnecessary repetition, the standard /? switch is not repeated for every command.

## Entering filenames

NOTE

Often, commands that accept filenames can accept either a simple filename, a filename that includes the file path, a filename that includes a drive name, or a fully qualified filename that includes drive and path and filename. The following examples show the EDIT command with some of these filename variations:

```
edit 123.bat

edit \batfiles\123.bat

edit d:123.bat

edit d:\batfiles\123.bat.
```

The syntax used to denote this filename flexibility is shown in this example:

[[*drive:*][*path*]*filename.ext*]

The square brackets indicate that each component is optional. The italics indicate that a component is a *variable*.

Because a filename is optional in the EDIT command, the filename components are enclosed in a pair of square brackets. DEVICE, on the other hand, requires a filename to be specified, that is, it is not optional, so it has the following syntax:

[*drive:*][*path*]*filename.ext*

## Entering wildcards

Some DOS commands operate on one or more than one file. A good example is the COPY command, which you can use to copy a single file or a collection of files. You can use the DOS wildcard characters * and ? to identify a filename pattern. The syntax for a command that accepts a filename or a wildcard filename is as follows:

[*drive:*][*path*][*wildcard.ext*]

Examples of some filenames that adhere to this command syntax are as follows:

```
*.*

c:\*.sys

d:\batfiles\????it.bat

config.sys

\windows\win.ini
```

## Complex commands

Some commands have rarely used switches that can complicate the syntax and the command description. In these cases, you will find an extra section called "Extended use." A good example of this treatment is with the [/A] and [/B] switches in the COPY command. Because you use the COPY command without these switches 99 percent of the time, they are discussed only in the "Extended use" section.

## Revisiting EDIT

The syntax for the EDIT command hopefully makes more sense now. You may recall that the syntax for EDIT is described as follows:

```
edit [[drive:][path]filename.ext] [/b] [/g] [/h] [/nohi]
```

To summarize, you can enter the EDIT command with no parameters, or you can optionally accept a single filename with as many as four switches.

## Command Summary

There are more than 75 commands and device drivers available in DOS 6. Appendix A provides an alphabetical summary of every command and a brief description of the command.

The appendix also identifies the commands that DOS 5 supports but that have been dropped (thankfully) from DOS 6. DOS 6 users can obtain the DOS 5 versions of these commands on a Supplemental disk from Microsoft Corporation or from DOS resellers.

BOB'S SCORECARD
5

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **ANSI.SYS** | ☐ | ☐ | ☐ | ■ |

### Description

The ANSI.SYS device driver provides ways to spruce up the DOS display capabilities, control the cursor location, and assign special functions to keys.

### Usage

```
device[high]=[drive:][path]ansi.sys [/x] [/k] [/r]
```

### Options and switches

```
[drive:][path]
```

This option identifies the directory in which ANSI.SYS is located, if it is not in the boot directory. SETUP places ANSI.SYS in the \DOS directory by default.

```
[/x]
```

On extended keyboards, you can perform some operations by using more than one physical key. Two keys can enter the + character, for example, and you can use two PgUp keys. The /X switch instructs DOS to treat these keys independently. With this switch in effect, you can remap each of a pair of keys differently.

```
[/k]
```

This switch instructs ANSI.SYS to treat an extended keyboard (with 101 keys) like a standard 84-key keyboard. In this state, ANSI.SYS treats an extended key the same as its standard equivalent.

```
[/r]
```

This switch adjusts line scrolling to make ANSI.SYS more compatible with programs designed for the disabled, which read the text from the display.

## Remarks

Having installed ANSI.SYS, you can use special escape sequences to manage the display colors, control the cursor location, and assign strings to keys. You can use the DOS commands PROMPT, ECHO, and TYPE to communicate the escape sequences to ANSI.SYS.

ANSI.SYS provides a way to add some spice to the boring DOS display, but Windows it isn't. ANSI.SYS gained its popularity in the days before graphical user interfaces existed, and by today's standards its capabilities are limited.

When ANSI.SYS is installed, you can use the MODE LINES= command to set EGA and VGA systems in condensed display mode with 43 or 50 lines.

Because the /X and /K switches are opposite instructions, you cannot specify both of them at the same time.

## Example

You make the following entry in the CONFIG.SYS file to install ANSI.SYS:

```
device=c:\dos\ansi.sys
```

## Additional information

Refer to Chapter 6 for a full discussion of ANSI.SYS, and to Appendix C for a summary of the escape codes that ANSI.SYS supports.

*See also* MODE.

## Bob's Shareware Collection disks

An excellent replacement for ANSI.SYS is FANSI-CONSOLE, from Hersey Micro Consulting, Inc. See Chapter 6 for further information about this utility.

# APPEND

BOB'S SCORECARD 3

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ■ | ☒ | ☐ |

## Description

APPEND is to data files what PATH is to programs. You use APPEND to tell DOS which drives and directories to search in for data files (such as spreadsheets and word processing documents) when a program doesn't find them in the default directory.

## Usage

```
append [[drive:][path1]] [;[drive:][path]] [[/x] or [/x:on] or
[/x:off]] [[/path:on] or [/path:off]] [/e]
```

## Options and switches

You can use APPEND with no switches. When APPEND is used in this way, DOS displays the active list of appended directories.

```
[[drive:][path1]] [;[drive:][path]]
```

Semicolons separate the list of directories to be searched for data files. Each directory can optionally include a drive identifier.

```
[[/x] or [/x:on] or [/x:off]]
```

The /X switch instructs DOS whether to search for programs and data files in the appended directories. The /X:OFF switch indicates that DOS should not search the appended directories for programs and batch files. /X and /X:ON have the same meaning, and both switches instruct DOS to search the appended directories for programs and batch files. To activate the /X (or /X:ON) status, you must specify the switch the first time you enter the APPEND command. You can use /X:OFF and /X:ON thereafter to toggle the search status.

```
[[/path:on] or [/path:off]]
```

Typically, when a program prompts you for a filename, you can enter either a basic filename and extension or a fully qualified filename including a path and even a drive letter. You use the /PATH switches to control whether appended directories are searched when you have entered a filename that includes a path. /PATH:ON specifies that appended directories be used even when you have entered a fully qualified filename. /PATH:OFF specifies that appended directories are searched only when you have entered a filename and no path.

```
[/e]
```

This switch instructs DOS to insert an APPEND entry in the environment. APPEND normally stores the list of appended directories internally, and this list is referenced by only low-level DOS functions, which in turn are called by general programs. By adding a copy of the APPEND directories to the environment, the appended directories are directly accessible to all programs and batch files.

You cannot use the /E switch in conjunction with the [[*drive*:][*path*] parameters. If you want to have an APPEND environment entry, you must use the APPEND command twice — first with the /E switch and then by specifying the appended directories. If you use APPEND first without the /E switch, the /E switch cannot be turned on later; therefore, you must reboot the system before executing APPEND with the /E switch.

To cancel the list of appended directories, enter the APPEND command followed by a space and a semicolon (`append ;`). A programmer must have dreamed up this syntax — why can't we use something as simple as APPEND /OFF?

## Remarks

APPEND is not the most important DOS command you will encounter. Because most contemporary programs provide their own ways to find data files, APPEND is really useful with only less sophisticated programs. Generally, good file and path management help you avoid the need for the APPEND command.

If you enter the APPEND command twice, the paths entered with the first command are overwritten by the second command.

DOS ignores any invalid paths entered with APPEND (no error message is generated).

## Examples

The following two APPEND commands instruct DOS to keep a copy of the appended directories in the environment and to identify three directories to be included in the appended list:

```
append /e
append c:\batfiles;c:\;c:\wp\docs
```

You can use the /X:OFF switch to instruct DOS to ignore the suspended directories as follows:

```
append /x:off
```

The following command assigns two appended directories and specifies that they should be ignored when a fully qualified filename is entered:

```
append d:\data\ap;c:\123r24 /path:off
```

## Additional information

*See also* PATH and SUBST.

Most programs use DOS services behind the scenes to perform common tasks such as creating, changing, and deleting files. These programs are serviced by calling an interrupt — the program interrupts DOS and asks for some data or service. One of the most common interrupts used by software is Interrupt 21h, which provides many file-related services. DOS searches appended directories for a file for the Open File (subfunction Fh), Open File Handle (subfunction 3Dh), and Get File Size (subfunction 23h) operations, even if the /X:OFF setting is in effect.

When /X:ON is in effect, the operations Find First Entry (subfunction 11h), Find First File (4Eh), and Execute Program (function 4Bh) search the appended directories.

Most users think that, after a command has been entered, DOS determines whether the command is internal. If not, they believe that DOS looks in the current directory for the program or batch file, and, finally, every directory in the environment PATH is searched. In fact, before the PATH is searched, DOS searches the *appended* directories if the /X:ON switch is active. This feature provides a convenient way to use APPEND to add one or more directories temporarily to the search path. If, for example, you enter the following command:

```
append c:\batfiles /x:on
```

DOS searches the BATFILES directory and looks for a program or batch file before searching the environment path. Furthermore, because the BATFILES directory is appended, you do not have to be in the BATFILES directory to edit a BAT file. You can edit a batch file called 123.BAT, for example, from any directory by simply entering the following command:

```
edit 123.bat
```

Because the appended directories are searched for programs and data files, you can then execute the batch file from any directory by entering the command

```
123
```

CAUTION Microsoft does not recommend using APPEND with Microsoft Windows or the Windows SETUP program.

# ATTRIB

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

## Description

Every file has four standard attributes: read-only, archive, system file, and hidden file. You use the ATTRIB command to set or clear any of these attributes for one or more files. You can also use ATTRIB to view file attributes.

## Usage

```
attrib [+r or -r] [+a or -a] [+s or -s] [+h or -h]
[[drive:][path]wildcard.ext] [/s]
```

## Options and switches

ATTRIB is the equivalent of entering the following command:

```
attrib *.*
```

ATTRIB can be entered with no parameters or switches; it displays by default the attributes of all the files in the default directory.

`[+r or -r]`

This option sets or clears the read-only attribute.

`[+a or -a]`

This option sets or clears the archive attribute.

`[+s or -s]`

This option sets or clears the system attribute.

`[+h or -h]`

This option sets or clears the hidden attribute.

`[[`*drive:*`][`*path*`]`*wildcard.ext*`]`

This option specifies a filemask that identifies the files whose attributes will be displayed or updated. The filemask must include a filename portion. To change the attributes in the BATFILES directory, for example, use the format `c:\batfiles\*.*` rather than `c:\batfiles`. If you don't specify a filename portion, ATTRIB addresses the specific directory entry rather than all the files in the directory.

`[/s]`

This switch instructs DOS to process all matching files in the current directory and all its subdirectories.

## Remarks

The most commonly used attributes are archive and read-only. The backup program, including DOS's own MSBACKUP command and all commercial backup utilities, normally clears the archive attribute when a file is backed up (saved to disk or tape). Whenever a file is revised, DOS resets the archive attribute.

You can use the read-only attribute to help prevent a file from being modified or deleted. That is, the file can only be *read,* not *written.* You can set the read-only attributes of the AUTOEXEC.BAT and CONFIG.SYS files to help prevent someone (or something) from modifying the file. The technique is far from foolproof, however. A savvy user (such as yourself) can use the ATTRIB command to remove the read-only attribute and edit and save the file, and then use ATTRIB to reactivate the read-only attribute.

You cannot delete files that have the read-only, system, or hidden attributes set. To delete this type of file, you first must turn off the S, H, and R attributes. If the following message is displayed when you try to delete a file:

```
Access denied
```

the file probably has one of these attributes set.

If a file has both the system and hidden attributes set, you can clear just one or the other but must clear them both at the same time.

## Examples

You can use the following command to make the AUTOEXEC.BAT file read-only:

```
attrib +r c:\autoexec.bat
```

To make sure that all the database files (with a DBF extension) can be deleted from the OLDFILES directory, enter the following command:

```
attrib -r -h -s c:\oldfiles\*.DBF
```

To review the file-attribute settings of all files in the current directory, enter the following command:

```
attrib
```

### Bob's Shareware Collection disks

The HotDIR Plus program, from Robert Woeger, displays file attributes when it is used with the /1 and /2 switches. PocketD from Pocketware can list and modify file attributes.

### Additional information

DOSSHELL can be used to change file attributes. The MSBACKUP, RESTORE, and XCOPY commands access the archive attribute.

In versions of DOS before 5.0, the DIR command did not display hidden files (files with the hidden attribute on) or search subdirectories for files. If you work on a system with an older version of DOS and you want to list an entire directory, including hidden files, simply use ATTRIB in place of DIR. Hidden files and standard files are listed.-

The /S switch instructs DOS to access all the files in the specified or default directory and all the subdirectories underneath it. This switch provides a quick and easy way to search an entire disk for one or more files, if the version of DOS does not support the DIR /S switch. If you cannot remember the location of the file RESUME.DOC, for example, you can enter the following command:

```
attrib c:\resume.doc /s
```

DOS then displays the attributes of *all* RESUME.DOC files on your hard disk. Along with the file attributes, DOS displays the directory in which each file was found.

## BACKUP

The BACKUP command was discontinued with the release of DOS 6. Refer to MSBACKUP for information about the new DOS backup command.

# BREAK

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ■ | ☒ | ■ |

## Description

If you ever have executed a command by mistake, such as running CHKDSK on the wrong drive, you probably have tried to abort the command by pressing Ctrl-Break or Ctrl-C. You use the BREAK command to control how frequently DOS checks to see whether you have pressed Ctrl-Break or Ctrl-C while a command or program is executing.

## Usage

You can either include the BREAK command in the CONFIG.SYS file or enter it from the command line. The syntax in the CONFIG.SYS file is as follows:

`break=on` or `break=off`

Except in the CONFIG.SYS file, the usage rules are as follows:

`break [on or off]`

## Options and switches

When you use the BREAK command with no options or switches, DOS displays the current BREAK setting.

As you might expect, the ON switch means that the extra checking for Ctrl-Break is enabled, and the OFF switch is used to disable the extra checking.

## Remarks

The BREAK setting is off by default, and DOS checks for Ctrl-C or Ctrl-Break only while getting user input from the keyboard or when writing to the screen or printer. When BREAK is set on, DOS additionally checks for Ctrl-C or Ctrl-Break when reading and writing to disk.

Enabling extended break-checking slows system performance somewhat, but not significantly.

## Examples

You can add the following line to the CONFIG.SYS file to activate extended Ctrl-C or Ctrl-Break checking:

```
break=on
```

Alternatively, you can add a line to the AUTOEXEC.BAT file as follows:

```
break on
```

To display the active setting, enter the command with no parameters as follows:

```
break
```

### Additional information

GEM

Most users think that pressing Ctrl-C is the same as pressing Ctrl-Break, but a DOS Jedi knows better.

Ctrl-Break is more reliable because it uses a special interrupt mechanism. Ctrl-C, on the other hand, is processed through the keyboard buffer like any other key combination. Although DOS's programs have been written to check the keyboard buffer for Ctrl-C, they look at only the first character in the buffer. If you have pressed another key before Ctrl-C, the DOS program does not notice it.

You can use the CHKDSK command to demonstrate this process. Execute CHKDSK and, while it is running, press Ctrl-C. The command terminates before completion. Run CHKDSK a second time, and press the spacebar and then Ctrl-C. CHKDSK notices only the spacebar in the keyboard buffer and therefore completes normally. Ctrl-Break gets the attention of CHKDSK regardless of the other keystrokes already waiting in the keyboard buffer.

## BUFFERS

BOB'S SCORECARD 4

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This command, designed to improve disk performance, controls the number of disk buffers used by DOS when reading files. Buffers are memory blocks designed to store data during disk read and write operations and to improve system performance.

### Usage

```
buffers=p[,s]
```

## Options and switches

When BUFFERS is not used, DOS assigns a default number of buffers. The value assigned depends on the amount of available memory, as follows:

| *Memory installed* | *Default buffers* |
|---|---|
| Less than 128K | 3 or less |
| 128K – 255K | 5 |
| 256K – 511K | 10 |
| More than 512K | 15 |

`=p`

This parameter specifies the number of (primary) buffers and must be in the range from 1 through 99.

`[,s]`

This option specifies the number of secondary or read-ahead buffers and must be in the range from 1 through 8. If S is not specified, the number of secondary buffers defaults to 0.

## Remarks

DOS buffers are similar to dumb disk caches. Whenever data is read from disk, it is transferred from the disk to the buffers and then from the buffers to main memory to be accessed by the executing program. If a program requests from disk some data that already is stored in a buffer, DOS transfers the data directory from the buffer to main memory. Memory-to-memory transfers are much faster than disk-to-memory transfers.

The secondary buffer cache is used as a *read-ahead* buffer — when data is read from disk, the sectors that follow are read also, in anticipation that this data also will be needed momentarily. Reading several sectors at a time is more efficient than reading them one at a time as necessary.

Many modern hard drives already do this kind of buffering on their controllers, so you will want to test to see if secondary buffers help or hinder performance in your system. Using the secondary buffer cache is especially useful on older systems. SMARTDRV.EXE doesn't work with the 8086 processor, and older hard drives do not have exotic hardware caches.

The value of BUFFERS has clearly diminished, thanks to superior disk-caching software, such as SMARTDRV. DOS, buffers and disk caches perform a similar function and, in some cases, can compete with each other and adversely affect system performance. If you use a disk cache, you should deliberately use a low buffer setting (2 or 4, for example) with no secondary buffers, to enable the disk cache to work unimpeded. You may want to use a higher BUFFERS setting (10, for example) if the disk-cache software does not support all drives, such as floppies or CD-ROM drives.

If you do not use a disk cache (and clearly you should), you should take advantage of BUFFERS. The default values are conservative, to say the least. Typically, a BUFFERS value of 20 to 40 is appropriate, and the secondary buffer cache should be set to its maximum value of 8. If too many buffers are allocated (more than 60, for example), DOS may spend more time managing the buffers than it would save by going directly to the disk. Also, more buffers mean less memory for your programs.

On a standard system, every buffer consumes approximately 530 bytes of memory; 512 bytes of this amount are used to store a single sector of data, and the remainder is used for buffer management. When DOS is loaded into the high memory area (HMA), the buffers also are loaded there, if room is available. After DOS has been loaded, room is available for 48 buffers. If you exceed 48, *all* the buffers are loaded into base memory. You can find out how much memory DOS is using for buffers by typing `mem /d/p`.

## Examples

Use the following command to allocate 40 standard buffers and no secondary buffers:

```
buffers=40
```

To allocate 20 buffers and 8 secondary buffers, use the following command:

```
buffers=20,8
```

## Additional information

*See also* MEM and SMARTDRV.

BOB'S SCORECARD 8

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **CALL** | ☐ | ☒ | ■ | ☐ |

## Description

If you use batch files, you will want to understand and use CALL. It enables one batch file program to execute a second batch file, after which control is returned to the first batch file.

## Usage

call [*drive:*][*path*]*filename* [*batch–parameters*]

## Options and switches

[*drive:*][*path*]*filename*

This option is the name of the batch file to be executed. The BAT extension does not have to be specified. CALL can be used to run COM and EXE files, in fact, but you have no good reason to do so. If you do not specify a file path, DOS searches for the file in the current directory and then in the path in the normal manner.

[*batch–parameters*]

If the batch file being called requires some parameters, you should specify them after the batch filename in the same way as you enter them from the command line.

## Remarks

The main value of CALL is that control returns to the calling program after the secondary batch file has finished executing. If you enter just the name of the second batch file (and do not precede the name with CALL), control is passed to the second batch file, and any remaining commands in the original batch file are *not* executed.

CAUTION

Do not use redirection symbols (<, >, <<, >>) and pipes (|) with the CALL command.

CALL enables a batch file to call itself, but you should provide some logical way for the operation to terminate; otherwise, it will loop forever. In fact, because each CALL execution consumes 80 bytes of conventional memory, eventually a looping batch file will terminate with an insufficient memory error.

*Note:* On older systems, before DOS 3.3, you may encounter a batch file statement beginning with `command /c`. Old Jedis used this trick before CALL was available. It has almost the same effect as CALL, but it uses a secondary copy of COMMAND.COM. Using the `command /c` trick, however, does not allow the second batch file to alter the environment permanently because it's acting on a copy, not the master environment. Unless you are writing batch files to run on pre-DOS 3.3 systems, use the CALL command.

### Examples

The following extract from an AUTOEXEC.BAT file shows the CALL command used to execute the batch file SAVEM.BAT:

```
@echo off
path c:\;c:\windows;c:\batfiles;c:\pckwik
call savem
prompt $p$g
```

To execute the batch file DOWNFONT.BAT and pass two parameters, you should add the following command to a batch file:

```
call downfont times helv
```

### Additional information

Refer to Chapter 10 for more information about batch file programming.

### Bob's Shareware Collection disks

The disk includes a number of tools for enhancing batch file operations: TurboBat, in the ExtraDOS Toolbox, compiles batch files; BATUTIL, from CtrlAlt Associates, provides slick user-interface tools.

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

## CD or CHDIR

### Description

This command changes the default (current) directory. Although CD is a basic, no-frills command, you probably use it every day. If it were taken away, you would surely miss it!

### Usage

```
cd [[drive:][path] or [..]]
```

## Options and switches

When you use CD with no options or switches, DOS displays the current default directory.

*[drive:][path]*

This option identifies the drive whose default directory will be changed, along with the new directory or path. You can omit the `drive:` component if you are changing the default directory of the current drive — as is usually the case. If the path does not commence with a backslash, the path is assumed to extend from the current directory.

`[..]`

Use a pathname of `..` to change to the parent directory. You can precede the two dots with a drive letter to change the default directory of another drive to the parent directory.

## Remarks

The CD command is one of the most commonly used DOS commands. To master DOS, you must be able to navigate quickly around directories. Remember that you can use the TREE command to display a visual map of a disk's directory structure.

## Examples

Assume that your hard disk is configured with the following structure:

```
C:\
│  LOTUS
│  ├─BUDGET
│  └─SAMPLES
├─WP
├─GAMES
├─PARADOX
├─BATFILES
├─WINDOWS
│  ├─SYSTEM
│  ├─NDW
│  └─WAVES
│     └─WAYNEWLD
└─DOS
```

To change directories from the root directory to the WINDOWS subdirectory called WAVES, you enter the following command:

```
cd\windows\waves
```

To *then* change directories to the subdirectory WAYNEWLD, you can enter one of the following commands:

```
cd waynewld

cd\windows\waves\waynewld
```

If the default directory is C:\LOTUS\SAMPLES, the following command changes the default directory to C:\LOTUS:

```
cd..
```

If the default directory is C:\LOTUS\SAMPLES, the following command changes the default directory to C:\LOTUS\BUDGET:

```
cd..\budget
```

### Additional information

*See also* PROMPT, RD, MD, and TREE.

### Bob's Shareware Collection disks

Because CD has not been enhanced significantly since an early version of DOS, a number of Shareware alternatives are smarter and easier to use than CD. My favorite is Keith Ledbetter's LCD utility. LCD enables you to change directories quickly to any drive on the system by entering a partial directory. If more than one directory matches the partial name, a pop-up dialog box enables you to choose the directory. Great!

## CHCP

BOB'S SCORECARD 3

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ■ | ☒ | ☐ |

### Description

The CHCP command is used with international character sets to display or change the active code page.

### Usage

```
chcp [nnn]
```

## Options and switches

When you use CHCP with no switches, DOS displays the current active code page.

[*nnn*]

This option specifies the number or ID of the code page to be activated.

## Remarks

Code-page management is necessary only when you want to change from the default code page to an alternative. It's not a command I use every day!

Unfortunately, changing code pages is not as easy as it seems. You can use CHCP to change code pages only after DOS has been primed with commands such as COUNTRY, NLSFUNC, and MODE.

Code pages are supported on the display, the keyboard, and the printer. Normally, you want all three devices to be on the same (code) page. CHCP sets the code page for all three devices. To change the code page on an individual device, use the MODE SELECT command.

Code pages are not supported on CGA and monochrome systems — the character sets are stored in video ROM and cannot be changed.

## Examples

Enter the following command to see the active code page:

```
chcp
```

If your system has been configured for Italian use, you can switch to the secondary Italian code page by entering the following command:

```
chcp 437
```

## Additional information

Refer to Chapter 9 of the *MS-DOS User's Guide* for a thorough discussion of international configuration issues.

*See also* COUNTRY, MODE, NLSFUNC, and KEYB.

BOB'S SCORECARD 7

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **CHKDSK** | ■ | ☒ | ☒ | ☐ |

## Description

The primary function of CHKDSK is to check the integrity of a disk's file structure. CHKDSK reports disk errors and, optionally, corrects them. CHKDSK displays a summary of the disk's use and the system's memory status.

## Usage

`chkdsk [`*drive:*`][[`*path*`]`*wildcard.ext*`] [/f] [/v]`

## Options and switches

When you use CHKDSK with no options or switches, DOS reports the status of the default drive.

`[`*drive:*`][[`*path*`]`*wildcard.ext*`]`

This option identifies the drive, filemask, or individual file whose integrity will be checked.

`[/f]`

This switch instructs CHKDSK to try to fix any errors it encounters.

`[/v]`

This switch specifies that a list of filenames should be generated as each file is checked.

## Remarks

The following example shows output generated by CHKDSK:

```
Volume Serial Number is 0000-16F5

   112 lost allocation units found in 9 chains.
    458752 bytes disk space would be freed

 211787776 bytes total disk space
     77824 bytes in 2 hidden files
    946176 bytes in 212 directories
```

```
206786560 bytes in 6432 user files
  3518464 bytes available on disk

     4096 bytes in each allocation unit
    51706 total allocation units on disk
      859 available allocation units on disk

   655360 total bytes memory
   528944 bytes free
```

The /F parameter is used to instruct DOS to correct any anomalies in the FAT. The most common problem encountered is lost allocation units. Although these clusters of the disk are marked as being in use, paradoxically no file claims to own them. In other words, they are not being used, but DOS does not reuse them. These situations may occur when a system crash occurs or when the system is turned off while a file is being saved to disk.

CHKDSK's solution to the problem is to remove the in-use flag for each of the clusters. This action frees up the clusters to be used again and provides more free disk space — something we all yearn for. Before clearing the in-use flag, DOS generates the following prompt:

```
112 lost allocation units found in 9 chains.
Convert the lost chains to files?
```

If you enter Y, the data from these clusters is saved, with a CHK extension, in a set of files in the root directory. You can peruse these files with an editor and see whether any data is worth salvaging. If you respond with N, the data is not saved.

Don't be overly concerned if one of the lines in the CHKDSK output identifies bad sectors. Most hard disks have some bad sectors, which are identified during the disk formatting process.

DOS 6

Previous versions of CHKDSK were a bit too bold about fixing things when the /F parameter was used. When two files each claimed some of the same clusters, CHKDSK /F would automatically arbitrate the dispute over the "cross-linked" clusters, sometimes with less than helpful results. DOS 6 modestly refrains from meddling with cross-linked clusters, even when /F is used. You will need a third-party file repair utility to perform this delicate surgery.

## Examples

To check the integrity of the default drive, enter the following command:

```
chkdsk
```

The following command checks for errors on drive C and corrects any errors it encounters:

```
chkdsk c: /f
```

The following command tests the integrity of a floppy disk in drive A:

```
chkdsk a:
```

## Additional information

Refer to Chapter 4 for more information on CHKDSK.

*See also* DEFRAG, MEM, and VOL.

Disk fragmentation can significantly impair the overall performance of a system. If a filename, or filemask, is specified with the CHKDSK command, DOS reports the overall disk status, along with any files (which match the filemask) that are fragmented.

The command `chkdsk *.*` entered from the WINDOWS directory, for example, generated the following output:

```
Volume Serial Number is 0000-16F5
Errors found, F parameter not specified
Corrections will not be written to disk

   101 lost allocation units found in 8 chains.
    413696 bytes disk space would be freed

 211787776 bytes total disk space
     77824 bytes in 2 hidden files
    946176 bytes in 212 directories
 206917632 bytes in 6433 user files
   3432448 bytes available on disk

      4096 bytes in each allocation unit
     51706 total allocation units on disk
       838 available allocation units on disk

    655360 total bytes memory
    528944 bytes free

C:\WINDOWS\AMILABEL.INI Contains 3 non-contiguous blocks
C:\WINDOWS\VBRUN100.DLL Contains 2 non-contiguous blocks
```

```
C:\WINDOWS\CONTROL.INI Contains 2 non-contiguous blocks
C:\WINDOWS\WINHELP.EXE Contains 5 non-contiguous blocks
C:\WINDOWS\FAXMNG.EXE Contains 3 non-contiguous blocks
C:\WINDOWS\PREC.EXE Contains 8 non-contiguous blocks
C:\WINDOWS\PREC.DOC Contains 2 non-contiguous blocks
C:\WINDOWS\PMIX.EXE Contains 10 non-contiguous blocks
C:\WINDOWS\MIX.EXE Contains 2 non-contiguous blocks
C:\WINDOWS\LSMTOGL.DLL Contains 2 non-contiguous blocks
C:\WINDOWS\FLW.INI Contains 2 non-contiguous blocks
C:\WINDOWS\WININI.OLD Contains 2 non-contiguous blocks
C:\WINDOWS\WBINXZ.EXE Contains 3 non-contiguous blocks
```

Refer to the DEFRAG command for information about how to defragment a disk.

CAUTION

CHKDSK expects to be the only program running. You should not run it from within a multitasking environment such as DesqView or Windows. When you run CHKDSK from such an environment, it often incorrectly reports allocation and fragmentation problems.

Also, CHKDSK does not work on drives that have been affected by the ASSIGN, SUBST, or JOIN commands.

Do not use the CHKDSK program from early versions of DOS on systems running DOS 6. Because many internal changes have taken place in DOS's file management, old versions of CHKDSK can corrupt the DOS 6 FAT.

## CHOICE

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 8 | ☐ | ■ | ■ | ☐ |

DOS 6

### Description

The command enables a batch file to prompt the user to select an option by pressing a key. CHOICE sets the ERRORLEVEL to indicate which key was pressed.

### Usage

```
choice [/c[:]keys] [/n] [/s] [/t[:]c,nn] [text]
```

### Options and switches

When you use CHOICE with no switches, DOS prompts the user to press Y or N.

`[/c[:]`*keys*`]`

This option identifies the list of valid keys the user can press. The colon is optional, but it should be used for clarity. If the /S switch is not specified, the valid keys are not case dependent (/abc and /ABC, for example, have the same meaning). Unfortunately, function keys and key combinations such as Alt-A are not supported.

`[/n]`

CHOICE displays the valid keys in square brackets by default ([A,B,C], for example). This display is referred to as the CHOICE prompt. Use /N to suppress the prompt display.

`[/s]`

The /S switch enables case sensitivity and instructs CHOICE to treat upper- and lowercase characters separately. When you do not specify /S, the key case is ignored.

`[/t[:]`*c*`,`*nn*`]`

This option instructs CHOICE to wait a specified number of seconds and then assume that a specific key was pressed. This switch is useful in batch files that occasionally can be run unattended. Again, the colon is optional, but it should be used for clarity. *nn* specifies the number of seconds to wait for the user to make a choice. If a key is not pressed after *nn* seconds, CHOICE continues as though the user has pressed the *c* character. *nn* must be a value in the range from 0 to 99. If a larger number is specified, the digits after the first two are assumed to be the prompt text. An error message is displayed if the character *c* is not one of the valid characters specified with the /C switch.

`[`*text*`]`

The remaining text following all the switches is echoed to the display before the character prompt. The text should be enclosed in quotation marks if it includes the / character.

## Remarks

You can determine which key the user pressed by checking the ERRORLEVEL returned by CHOICE. If the user pressed Ctrl-Break or Ctrl-C, a 0 error level is returned; if an error is in the command switches and CHOICE fails, a 255 ERRORLEVEL is returned; otherwise, the error level indicates the key the user pressed — an error level of 1 is returned if the first key (specified with `/c:keys`) was selected, a 2 for the second key, and so on. You can test the

ERRORLEVEL with the IF batch file command to divert execution to the appropriate section of the batch file.

Although you can execute CHOICE from the DOS prompt, you have no sane reason to do so.

The maximum number of characters in a single CHOICE command is 127, which provides a maximum text prompt of 120 characters when no switches are used. When additional switches are specified, the maximum text size is reduced accordingly.

## Examples

The following batch file uses all the default switches but specifies a brief message:

```
@echo off
CHOICE Do you want to load the mouse
if errorlevel 2 goto quit
mouse /y
:quit
```

When the batch file is executed, the valid keys default to *y, Y, n,* and *N,* and the following prompt appears:

```
Do you want to load the mouse [Y,N]?
```

The following batch file uses the /C switch to specify the valid keys:

```
@echo off
CHOICE /c:mtn Do you want to load the Mouse, Tablet or Neither
if errorlevel 3 goto end
if errorlevel 2 goto Tablet
mouse /y
goto end
:Tablet
tabk /c:345
:end
```

The valid keys are M, T, or N in upper- or lowercase. When the batch file is executed, the following prompt appears:

```
Do you want to load the Mouse, Tablet or Neither [M,T,N]?
```

The batch file tests the ERRORLEVEL (in reverse order) to ascertain which key the user pressed. A value of 3 indicates that the user pressed N, and a value of 2 indicates that the user pressed T; otherwise, the mouse driver is installed.

You can improve the CHOICE statement in the preceding batch file example by adding a /T switch as follows:

```
choice /c:mtn /t:n,9 Do you want to load the Mouse, Tablet or Neither
```

If the user does not make a choice within nine seconds, CHOICE returns an ERRORLEVEL of 3, as though the user pressed N.

## Additional information

CHOICE provides an easy way to build a menu system with a single batch file. Refer to Chapter 10 for more information about this technique and other ways to use CHOICE.

*See also* ECHO, IF, and PAUSE.

## Bob's Shareware Collection disks

Although CHOICE *really* improves the power of batch files, it pales in comparison to STACKEY and BATUTIL. The Turbobat batch file compiler, from Foley Hi-Tech Systems, provides an excellent way to stop users from tampering with batch files and improves batch file performance.

Do not use the pipe and filter characters (|, <, >, >>, <<) in the CHOICE text, even if the text is enclosed in quotation marks. DOS interprets these characters as command switches and causes CHOICE input or output to be suppressed.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **CLS** | 5 | ☒ | ■ | ☐ | ☒ |

## Description

The CLS command is used to clear the screen. CLS removes all the displayed characters, sets the color to gray on black, and repositions the cursor to the top left corner of the display.

## Usage

```
cls
```

## Options and switches

None

### Remarks

Although you can use CLS from the DOS prompt, it most often is used in a batch file before executing a command, displaying a message, or drawing a menu.

You can use ANSI.SYS to customize the display colors you may prefer to gray on black.

### Example

The following batch file fragment uses CLS to clear the screen and then displays a message:

```
.....
cls
echo Make sure a floppy disk is in drive A.
echo To abort the operation press Ctrl-Break.
pause
.....
```

CROSS REFERENCE

### Additional information

*See also* ANSI.SYS.

DISK

### Bob's Shareware Collection disks

The BATUTIL batch enhancer, from CtrlAlt Associates, provides a variety of tools for enhancing your display.

BOB'S SCORECARD 5

## COMMAND

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ☒ | ☒ | ■ |

### Description

Starts a new copy (or instance) of COMMAND.COM, the command interpreter. As its name suggests, the command interpreter is responsible for accepting user input commands and invoking the appropriate function or external command.

Although the COMMAND.COM file is an essential part of DOS, you rarely need to execute the command directly. You can use COMMAND directly to adjust the environment size, but the preferred technique is to use the SHELL command.

## Usage

The syntax for COMMAND is as follows:

```
command [[drive:]path] [device] [/e:nnnnn] [/p [/msg]] [/c text]
[/k filename]
```

When COMMAND is used with the SHELL command, the syntax is as follows:

```
shell=[[drive1:]path1]command.com [[drive:]path] [device] [/e:nnnnn]
[/p [/msg]]
```

## Options and switches

Used with no options or switches, COMMAND invokes a secondary copy of COMMAND.COM. You can unload this secondary copy by entering the EXIT command. Although this command seems to be of little value, it provides the opportunity for some clever tricks (see the upcoming SECRET).

```
[[drive1:]path1]command.com
```

This option identifies the location of the command interpreter (COMMAND.COM) when it is not located in the root directory of the boot drive.

```
[[drive:]path]
```

This option specifies where DOS should look for COMMAND.COM during a session. This path is used to update the environment variable COMSPEC.

```
[device]
```

This option identifies a DOS device to be used for default command input and output. Normally, the device is CON (the console or monitor). The following table shows the valid devices.

| ***Device*** | ***Description*** |
|---|---|
| AUX | The auxiliary device |
| CON | The console or display |
| LPT1 | The device connected to LPT1 |
| LPT2 | The device connected to LPT2 |
| LPT3 | The device connected to LPT3 |
| PRN | The printer |
| COM1 | The device connected to COM1 |
| COM2 | The device connected to COM2 |
| COM3 | The device connected to COM3 |
| COM4 | The device connected to COM4 |

Under normal circumstances, you do not need to use this switch. These capabilities usually are exploited by specialist programs involving remote operations, such as debuggers and file-transfer programs.

***Note:*** CTTY can identify a different input-output device without invoking a secondary copy of COMMAND.COM.

```
[/e:nnnnn]
```

This switch is used to specify the size of the environment, which can range from 160 to 32,768 bytes. The switch is commonly used when COMMAND is invoked with the SHELL statement in CONFIG.SYS. If the switch is not used, the environment size defaults to 256 bytes.

```
[/p [/msg]]
```

The /P switch makes the new copy of COMMAND.COM permanent. That is, you cannot EXIT back to the parent command processor. When the /P switch is used, DOS automatically executes AUTOEXEC.BAT. Normally, you should use this switch only when COMMAND is invoked with the SHELL statement in CONFIG.SYS. When the /P switch is used, the /MSG switch can be specified to instruct DOS to keep all DOS messages in memory. The /MSG switch is designed for unfortunate users who are running DOS from floppy disks — messages can be displayed without retrieving them from disk every time.

```
[/c text]
```

Before the introduction of the CALL batch command, the /C switch was used in batch files to launch other batch files. The TEXT command identifies the command or program to be executed as soon as the secondary copy of COMMAND.COM is loaded. Note that a space is required between the /C and the command text.

```
[/k filename]
```

This switch instructs COMMAND to execute the specified program and leave the user at the DOS prompt when the program has executed. This switch was introduced with DOS 6 primarily to enable Windows users to execute a start-up batch file other than AUTOEXEC.BAT when selecting the DOS Prompt window.

You should not use this command with the SHELL statement.

## Remarks

In versions of DOS before 3.3, COMMAND was often used as a way to run secondary batch files; since 3.3, however, the CALL command performs that task. Nowadays, COMMAND is used primarily with the SHELL statement in CONFIG.SYS to adjust the environment size.

To economize on memory use, the DOS command interpreter is loaded into memory in two blocks. The *resident* block is loaded permanently in memory and remains there for the entire session. The code that interprets user input from the DOS prompt, however, is loaded in the top of base memory and can be overwritten by an executing program. After all, the executing program does not need access to this code. This part of COMMAND.COM is referred to as the *transient* block. When the program terminates, DOS reloads the transient block from COMMAND.COM.

### Examples

The following line from CONFIG.SYS uses COMMAND with the SHELL statement to increase the environment size to 500 bytes. It also sets the COMSPEC environment variable to C:\DOS6\COMMAND.COM:

```
shell=c:\dos6\command.com c:\dos6 /e:500
```

The following line from a batch file uses the pre-DOS 3.3 trick of using a secondary copy of COMMAND.COM to launch a batch file:

```
command /c saveit.but
```

### Additional information

CROSS REFERENCE

Refer to Chapter 10 for more information on using the /K switch with COMMAND.

*See also* CALL, CTTY, EXIT, PATH, and SHELL.

GEM

Sometimes I want to change my path statement temporarily so that DOS searches for programs in an alternative set of directories. On my own system, I use PutPath and GetPath batch files; when I work on someone else's system, these files are not always available.

A nifty way to change the path temporarily is to execute COMMAND before changing the path. After I finish experimenting, I return to the original path by entering EXIT. This step terminates the secondary copy of COMMAND.COM along with the modified environment and restores the original environment.

## COPY

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ■ | ☐ |

### Description

This command copies one or more files to another location or to the same location with a new filename. You can instruct COPY, in its extended form, to treat the files as ASCII or binary.

## Usage

```
copy [drive:][path]wildcardS.ext [drive:][path][wildcardT.ext] [/v]
```

## Options and switches

*[drive:][path]wildcardS.ext*

This option identifies the source file or files to be copied. When a path is specified with no filename (C:\DOS, for example), DOS assumes the *.* wildcard. You must specify this switch.

*[drive:][path][wildcardT.ext]*

This option identifies the target location where the files will be copied. If no filename is specified, the files retain their original names. If a wildcard is used, the file copies are renamed by using the wildcard. If a single target file is specified with multiple source files, the source files are concatenated (combined) in a single target file. If no target path or filename is specified, the source files are copied to the current directory.

`[/v]`

This switch instructs DOS to verify that the data in the target file can be read successfully. It does not check to see whether the target is an accurate copy of the source. Unless you are experiencing unreliable disk access, you do not need to use this switch. COPY operations take about 30 percent longer when this switch is active.

## Remarks

The source or target filenames can be devices. You may be familiar with the following statement:

```
copy con newfile.ext.
```

This command instructs DOS to copy characters from the console (or keyboard) to the file NEWFILE.TXT. Any characters you type are added directly to the file. This command provides a quick way to create a file when no editor is available. You should enter the end-of-file character Ctrl-Z to close the file and return to the DOS prompt. Similarly, you can print a file by copying it to a printer device (PRN or LPT1, for example).

COPY does not copy zero-length files, although XCOPY does. COPY does not copy hidden or system files, nor does XCOPY. You must use the ATTRIB command first to remove the hidden and system file attributes.

The target file normally is assigned the same date and time stamp as the source, but the archive attribute is set to ON to ensure that the new file is included in the next incremental backup.

The read-only file attribute is not copied to the target file — the target file always has the read-only flag turned off.

## Examples

Use the following command to copy AUTOEXEC.BAT to AUTOEXEC.BAK:

```
copy c:\autoexec.bat *.bak
```

To copy all the files with a BAT extension in the BATFILES directory to the OLDSTUFF directory, enter the following command:

```
copy \batfiles\*.bat \oldstuff
```

The following command copies all the files from the root directory to the current directory:

```
copy \
```

The following command prints the file INFO.TXT to the printer connected to LPT1:

```
copy info.txt lpt1
```

Be careful when you are using wildcards in the source and not in the target. The following command copies all the BAT files in the current directory and creates a single large file called BLOB:

```
copy *.BAT BLOB
```

## Additional information

CROSS REFERENCE

Refer to Chapter 2 for more information on COPY.

*See also* DEL, MOVE, RENAME, REPLACE, VERIFY, and XCOPY.

## Bob's Shareware Collection disks

DISK

PCOPY, one of the Patriquin Utilities from Patri-Soft, is a superb shareware utility for copying files. PCOPY can copy files based on date or size, "best-fit" files for optimal use of floppy disk free space, synchronize source and target disks, prompt before overwriting files, and process all nested subdirectories. After you use PCOPY, you will wonder how you got along without it.

PocketD from Pocketware also provides excellent file copying utilities.

## Extended use

In rare circumstances, you may need to use the + operator and the /A and /B switches with the COPY command.

The + operator is used to combine multiple source files in a single target file. The following command combines three files (FILE.ONE, FILE.TWO, and FILE.TRE) in a single large file, FILE.BIG:

```
copy file.one+file.two+file.tre file.big
```

DOS can use one of two ways to determine how many bytes to copy: by referencing the file's directory entry and noting the size of the file in bytes, or by copying bytes until the Ctrl-Z or end-of-file character is encountered. You can use the /A and /B switches with both the source and target files to control the way DOS manages the COPY operation.

The /A switch is designed for use with ASCII files that contain one Ctrl-Z character. Used with the source file, /A instructs DOS to copy bytes up to the first Ctrl-Z. Used with the target, /A instructs DOS to add a Ctrl-Z to the end of the target file.

The /B switch is used with binary files that may have no Ctrl-Z characters or many of them. /B instructs DOS to ignore Ctrl-Z characters and use the directory entry to determine the total file size.

**GEM**

You can use the COPY command to change the date and time of a file. The trick is to add a plus and two commas (+,,) after the source filename and remember to use /B. To update AUTOEXEC.BAT to the current date and time stamp, for example, change to the boot directory and enter the following command:

```
copy /b autoexec.bat+,,
```

The plus-comma-comma suffix can be used, when you are copying files, to instruct DOS to assign the current date and time to all the new target files.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **COUNTRY** | 3 | ☐ | ☐ | ☐ | ■ |

## Description

The COUNTRY command configures the country language conventions for your system.

## Usage

```
country = xxx [,yyy] [,[drive:][path[filename.ext]]
```

## Options and switches

*xxx*

This code identifies the country whose conventions are used. The code is the same as the standard international dialing directories.

*,yyy*

DOS uses as many as 256 different characters to display information about the screen and to send to the printer. A complete set of 256 characters is referred to as a *code page.* The optional *yyy* parameter identifies the code page DOS will use. Every country DOS supports has a standard code page and an alternative code page. If this option is not specified, DOS uses the standard code page associated with the specified country code.

[,[*drive:*][*path*[*filename.ext*]]

The syntax for the COUNTRY command is the opposite of most device drivers because the driver filename is specified after the other switches (perhaps in deference to the French). If the COUNTRY.SYS file is not located in the root directory of the boot drive, or if the device driver is not named COUNTRY.SYS, you must specify the fully qualified filename of the device driver. If SETUP was used to install DOS, COUNTRY.SYS is located in the \DOS directory.

## Remarks

If you are a U.S. user and you have no international systems-support issues, you don't have to worry about this command. It may come as a shock, but not everybody wants a PC configured for the U.S.A.

The COUNTRY.SYS device driver is used to configure a system in order to use another country's display conventions. Areas affected by COUNTRY.SYS are the date and time display, the characters in the alphabet, the symbol used to denote currency, and the sort order of files. If you do not have a COUNTRY entry in CONFIG.SYS, your system automatically is configured as a U.S.A. system.

To fully configure a system for international use, you should use the KEYB command to control the keyboard properties.

## Examples

You can use the following entry in CONFIG.SYS to set the country setting to Spain:

```
country=034
```

To configure the system to Italy when the device driver is located in the \DOS directory, enter the following command (note the two commas):

```
country=039,,c:\dos\country.sys
```

Use the following command to configure the system for Denmark with the alternative code page:

```
country=045,865,c:\drivers\country.sys
```

## Additional information

CROSS REFERENCE

Refer to Chapter 9 of the *MS-DOS User's Guide* for further information about international customization.

*See also* KEYB, NLSFUNC, DISPLAY.SYS, PRINTER.SYS, and CHCP.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **CTTY** | 3 | ■ | ☒ | ☒ | ☐ |

## Description

This command is an odd one. CTTY enables a different device to be used for keyboard and display I/O.

## Usage

`ctty` *device*

## Options and switches

*device*

This option, the alternative device source, can be any one of the following: PRN, LPT1, LPT2, LPT3, CON, AUX, COM1, COM2, COM3, and COM4.

## Remarks

CTTY is not your everyday DOS command. You will use it rarely, if ever. Its most common use is to access your computer remotely. It is used with the DOS 6 command INTERSVR, for example, when the /RCOPY switch has been specified to transfer the InterLink files to another computer using a serial port.

To use CTTY with a serial port, the MODE command first must be used to set the communication characteristics (baud rate and parity, for example).

To return control to the default console, specify the device CON.

The default console can also be reassigned by using COMMAND.

### Examples

Use the following two commands to prepare a client PC to be controlled by an INTERSVR on the server PC, ready for remote installation of INTERLNK:

```
mode com1:2400,n,8,1,p
ctty com1
```

### Additional information

*See also* COMMAND, INTERSVR, INTERLNK, and MODE.

## DATE

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 5 | ☒ | ☒ | ☒ | ☐ |

### Description

The DATE command sets the system date.

### Usage

```
date [mm-dd-[yy]yy]
```

### Options and switches

When DATE is executed without switches, it displays the current system date and prompts for the input of a new date. Just press Enter to return to the DOS prompt.

```
[mm-dd-[yy]yy]
```

This option specifies the new system date. The format and order of the days, months, and years depends on the country configuration. In the U.S., the standard format is mm-dd-yy, where the year is in the range from 80 to 99 (1980 to 1999). You can specify the year in 4-digit form to enter years in the range from 1980 to 2099. The days, months, and years separators can be periods (.), hyphens (–), or slashmarks (/).

### Remarks

You must keep the system date accurate so that filenames are assigned the correct date. Calendaring software also uses the system date.

### Example

The following command sets the date to my 40th birthday:

```
date 02-20-1996
```

### Additional information

CROSS REFERENCE

*See also* COUNTRY and TIME.

## DBLSPACE

BOB'S SCORECARD 9

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

The DBLSPACE command creates and manages compressed drives on hard disks and floppies.

### Usage

The following syntax lines are all acceptable uses of the DBLSPACE command:

```
dblspace

dblspace /chkdsk [/f] [drive:]

dblspace /compress drive1: [/newdrive=drive2:] [/reserve=size]

dblspace /convstac=stacvol drive1: [/newdrive=drive2:] [/cvf=sss]

dblspace /create drive1: [/newdrive=drive2:] [/size=sz or
/reserve=sz]

dblspace /defragment [drive:]

dblspace /delete drive:

dblspace /format drive:

dblspace [/info] drive:
```

```
dblspace /list

dblspace /mount[=nnn] drive1: [/newdrive=drive2:]

dblspace /ratio[=factor] [drive: or /all]

dblspace /size[=size1 or /reserve=size2] drive:

dblspace /unmount [drive:]
```

## Options and switches

When you use DBLSPACE with no switches, DOS displays a full-screen menu. Alternatively, you can execute DBLSPACE with a variety of switches to automatically invoke a specific DBLSPACE command.

The /NEWDRIVE switch, common to many of the DBLSPACE commands, is described as follows:

```
[/newdrive=drive2:]
```

When an entire standard DOS drive is compressed, an extra logical drive is created. The drive letter originally assigned to the uncompressed drive is reassigned to the compressed drive. The portion of the original drive that remains uncompressed is assigned a new drive letter. The next available drive letter is used by default. The /NEWDRIVE switch can optionally be used to instruct DBLSPACE to use a specific drive letter. The switch can be abbreviated to /NEW.

The main DBLSPACE commands and their related switches follow:

```
dblspace /chkdsk [/f] [drive:]
```

In the same way that CHKDSK checks the integrity of a standard DOS drive, the /CHKDSK switch instructs DBLSPACE to check the integrity of a DoubleSpace drive. The /F switch instructs DOS to correct any errors it encounters. If a drive is not specified, DBLSPACE checks the default drive.

```
dblspace /compress drive1: [/newdrive=drive2:] [/reserve=size]
```

This command instructs DBLSPACE to create a compressed DoubleSpace drive from the existing standard DOS drive *drive1:* and compress the files on the drive. Use the /RESERVE (or /RES) switch to control how much of the original drive is left uncompressed. By default, DBLSPACE reserves 2MB on a hard drive.

```
dblspace /convstac=stacvol drive1: [/newdrive=drive2:] [/cvf=sss]
```

This command converts an already compressed Stacker drive, *drive1:*, to a DoubleSpace drive. STACVOL identifies the Stacker volume to be converted. A DoubleSpace drive is a large hidden file with the name `dblspace.`*nnn*; you can use the /CVF switch (short for *c*ompressed *v*olume *f*ile) to specify a filename extension for this hidden file. The extension must be a three-digit number in the range from 000 to 254. By default, DBLSPACE assigns the lowest available (unused) number as the extension.

```
dblspace /create drive1: [/newdrive=drive2:] [/size=sz or
/reserve=sz]
```

This command creates an empty DoubleSpace drive by using the free space on standard DOS drive *drive1:*. Use /SIZE or /RESERVE to control how much space is given to the DoubleSpace drive. The /SIZE switch specifies in megabytes the amount of physical space that should be used to create the DoubleSpace drive. Conversely, /RESERVE specifies in megabytes how much physical space should not be used in creating the DoubleSpace drive (that is, how much free disk space to leave uncompressed). Decimal values are accepted (/SIZE=5.75, for example). If neither switch is specified, DOS assumes a switch of /RESERVE=2.

```
dblspace /defragment [drive:]
```

This command is DBLSPACE's equivalent of DEFRAG for DoubleSpace drives. If the drive is not specified, the default drive is used.

```
dblspace /delete drive:
```

Use this powerful switch with caution. The /DELETE switch deletes a DoubleSpace drive and all the files stored on the compressed drive. Kaboom. See the caution at the end of this command section.

```
dblspace /format drive:
```

Use this destructive command with caution. Like the standard DOS FORMAT command, /FORMAT formats a DoubleSpace drive and destroys every file stored on the drive. Unfortunately, because you cannot unformat a DoubleSpace drive, you have no chance to recover the formatted files. Ouch!

```
dblspace [/info] drive:
```

This switch displays interesting information about a DoubleSpace drive, including compression statistics and the drive label. If you execute the DBLSPACE command and specify only a DoubleSpace drive letter, DOS assumes the /INFO switch.

```
dblspace /list
```

This switch lists all the DoubleSpace drives on the system.

```
dblspace /mount[=nnn] drive1: [/newdrive=drive2:]
```

This command remounts a DoubleSpace drive that was previously disabled using the /UNMOUNT switch. You can use the *nnn* switch to specify the filename extension of the hidden DoubleSpace file. The extensions are three digits in the range from 000 to 254, and DOS tries by default to mount the hidden file DBLSPACE.000.

```
dblspace /ratio[=factor] [drive: or /all]
```

DOS reports the amount of free space available on a DoubleSpace drive by multiplying the physical free space by the compression ratio. The /RATIO switch updates the compression ratio used in this calculation. Any ratio in the range from 1.0 to 16.0 is accepted, but it normally is in the range from 1.5 to 2.5. If /RATIO is specified with no ratio factor, DOS uses the average compression ratio of all the existing files on the drive. You can use /RATIO on a single drive or on all drives.

At boot time, DOS automatically adjusts each DoubleSpace drive's compression ratio based on the compression ratio of each drive's existing files.

```
dblspace /size[=size1 or /reserve=size2] drive:
```

This command changes the size, in megabytes, of a DoubleSpace drive. Behind the scenes, DOS changes the total size of the hidden `dblspace.nnn` file. You can specify the size explicitly with the *=size1* argument, or, conversely, use the /RESERVE switch to tell DOS to leave the specified number of megabytes on the uncompressed (parent) drive.

```
dblspace /unmount [drive:]
```

This command makes a DoubleSpace drive unavailable temporarily and frees the drive letter assigned to it. It can be remounted with the /MOUNT switch.

## Remarks

CAUTION

Don't run DBLSPACE from within a multitasking environment such as DOSSHELL or Windows.

Both the /COMPRESS and /CREATE switches create new DoubleSpace drives. Use the /COMPRESS switch to compress all the files on an existing drive, and use /CREATE to create a new empty drive using the free space on an existing drive.

A standard DOS drive must have free space if it is to be compressed. A hard disk requires a minimum of 1MB of free space, and a floppy drive requires 200K.

If DBLSPACE encounters a Windows swap file on a drive that is being compressed, the file automatically is moved to the uncompressed drive.

You can use the /UNMOUNT and /MOUNT switches to assign a DoubleSpace drive a new drive letter. See Chapter 4 for more information.

You can use the file DBLSPACE.INF to control some of the default actions of DBLSPACE.

## Examples

Use the following command to start DBLSPACE in interactive mode and prompt you for selections and options:

```
dblspace
```

To create a new drive using 40MB of drive D, use the following command:

```
dblspace /create f: size=40.
```

Use the following command to create a new drive, compress all the files on drive C, and leave the minimum amount of space for the uncompressed drive:

```
dblspace /compress c: /reserve=0
```

To defragment the files stored on DoubleSpace drive F, enter the following command:

```
dblspace /def f:
```

## Additional information

CROSS REFERENCE

Refer to Chapter 4 for a thorough discussion of the DBLSPACE utilities.

*See also* CHKDSK, DBLSPACE.SYS, DEFRAG, and FORMAT.

CAUTION

The /DELETE and /FORMAT commands are destructive. Any existing files on the DoubleSpace drive are destroyed. Use these switches with caution, and make sure that you do not need the data on the drive.

If you accidentally delete a DoubleSpace drive, you can recover your files if you use UNDELETE immediately to restore the `dblspace`.*nnn* file.

You cannot recover DoubleSpace files that have been formatted accidentally. Be warned!

## DBLSPACE.SYS

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This command loads the disk-compression driver DBLSPACE.BIN to the bottom of base memory or upper memory.

### Usage

You should add the DBLSPACE.SYS statement to CONFIG.SYS and load it either in base memory with a DEVICE= statement or in upper memory with a DEVICEHIGH= statement:

```
device[high]=[drive:][[path]]dblspace.sys /move
```

### Options and switches

```
[drive:][[path]]
```

This option identifies the location of the DBLSPACE.SYS file. SETUP puts this file in the \DOS directory by default.

```
/move
```

The /MOVE switch is really a comment. Any switches specified with DBLSPACE.SYS are ignored. The developers of DOS added the /MOVE switch to provide a hint about the purpose of the device driver.

### Remarks

DBLSPACE.SYS is not a device driver for DoubleSpace drives — it is a device driver to *move* the DBLSPACE.BIN device driver in memory.

During system boot-up, IO.SYS loads DBLSPACE.BIN at the top of conventional memory and just leaves it there for the duration of the processing of CONFIG.SYS, or until DBLSPACE.SYS is loaded and moves it. If the DBLSPACE.SYS statement is added to the CONFIG.SYS with the DEVICE= statement, DBLSPACE.SYS will move DBLSPACE.BIN to the bottom of conventional memory. If the DBLSPACE.SYS statement specifies DEVICEHIGH, it is DBLSPACE.BIN that will be moved to upper memory.

During system boot-up, IO.SYS loads DBLSPACE.BIN before CONFIG.SYS has even been looked at. It initially is loaded at the top of memory, adjacent to the memory area occupied by IO.SYS. It must be loaded into base memory because no memory management software has been loaded to create the UMBs that early in the booting cycle. If the CONFIG.SYS file does not contain a DBLSPACE.SYS driver, DBLSPACE.BIN is left at the top of conventional memory. This can be a problem if one of your other drivers wants the use of the top of conventional RAM, too! Consequently, DBLSPACE.SYS should be declared early on, just after your memory manager, in your CONFIG.SYS so that it can do its work and free up the potentially disputed real estate at the top of the conventional memory.

### Example

Use the following command to instruct DOS to move DBLSPACE.BIN into upper memory with DBLSPACE.SYS:

```
devicehigh=c:\dos\dblspace.sys /move
```

### Additional information

CROSS REFERENCE

Refer to Chapter 4 for a thorough discussion of the DBLSPACE utilities.

*See also* DBLSPACE.

## DEBUG

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

This command starts the infamous DEBUG program for creating and analyzing program files.

### Usage

debug [[*drive:*][*path*]*filename*] [*parameters*]

### Options and switches

When you enter DEBUG with no switches, the terse hyphen prompt (–) is displayed. In this mode, you can execute debug commands interactively. Enter a ?, for example, to see a list of all the DEBUG commands.

[[*drive:*][*path*]*filename*]

This option identifies the name of the program to be debugged.

[*parameters*]

This option identifies any command-line parameters required by the program being debugged.

### Remarks

Enter `q` at the hyphen prompt to quit from DEBUG.

### Example

Use the following command to execute debug and load the file DEMO.EXE:

```
debug demo.exe
```

### Additional information

Refer to Chapter 11 for more information about using DEBUG.

## DEFRAG

BOB'S SCORECARD 9

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ■ | ☐ |

### Description

This command increases disk performance by defragmenting files.

### Usage

```
defrag [drive:] [/f or /u] [/s[:]order] [/v] [/b] [/skiphigh]
[/bw] [/lcd] [/g0]
```

### Options and switches

When you execute DEFRAG with no switches, a menu of choices appears. You can then press F1 to display DEFRAG's custom on-line help.

[*drive:*]

This option specifies the drive to be defragmented. If a drive is not specified, the default drive is used.

```
[/f or /u]
```

The /F switch (for *f*ull optimization) ensures that no empty spaces are between files. This switch helps to reduce future file fragmentation but increases the time taken to perform the defragmentation.

The /U switch instructs DEFRAG to defragment the files and leave any empty spaces (or "holes") that may be created.

`[/s[:]`*order*`]`

This switch instructs DEFRAG to reorganize the files and directories so that they are sorted in a specific order. The following switches are used with /S to identify the sort order:

| ***Switch*** | ***Sort Order*** |
|---|---|
| N | Sorted by filename in ascending order (A first) |
| –N | Sorted by filename in descending order (Z first) |
| E | Sorted by extension in ascending order |
| –E | Sorted by extension in descending order |
| D | Sorted in date/time order (oldest files first) |
| –D | Sorted in date/time order (most recent files first) |
| S | Sorted by file size (smallest first) |
| –S | Sorted by file size (largest first) |

The colon (:) in the /S command is optional but should be used for clarity.

```
[/v]
```

The /V switch instructs DEFRAG to verify the written data by performing a CRC check. This switch can slow the defragmentation process by as much as 30 percent.

```
[/b]
```

The /B switch instructs DEFRAG to reboot the computer on completion.

```
[/skiphigh]
```

This switch forces DEFRAG to load into conventional memory. By default, DEFRAG tries to load itself into UMBs and leave the maximum memory available for the defragmentation operation.

```
[/bw]
```

The /BW switch forces DEFRAG to use monochrome colors.

```
[/lcd]
```

Like /BW, the /LCD switch forces DEFRAG to use colors that can be viewed easily on LCD (laptop) displays.

```
[/g0]
```

DEFRAG uses custom characters by default to give the program a graphical look. Specify /G0 to force DEFRAG to use standard ASCII characters rather than the custom ones.

## Remarks

To make sure that no lost allocation units exist, before running DEFRAG, run the following command:

```
chkdsk /f
```

Run DEFRAG at least once a week to optimize your hard disk performance and to enhance your chances for data recovery.

You should not run DEFRAG inside Windows, DOSSHELL, or any other multitasking environment. DEFRAG can defragment only local drives, not network drives.

Because DEFRAG is licensed from Symantec, some discrepancies exist between the statistics reported by DEFRAG and the "same" statistics reported by CHKDSK. These discrepancies are not serious and reflect different reporting methodologies. DEFRAG includes the root directory in the count of total directories, for example, and CHKDSK does not.

If DEFRAG fails, an error code is set that can be checked by an `if errorlevel` statement in a batch file. The following ERRORLEVELs can be generated:

| *ERRORLEVEL* | *Message* |
|---|---|
| 0 | All is well, DEFRAG completed with no errors. |
| 1 | An unexpected internal error occurred. |
| 2 | DEFRAG could not run because no free clusters existed. Create some space by deleting unnecessary files, or move some files temporarily to another disk. |
| 3 | Ctrl-Break was pressed to abort the operation. |
| 4 | A general error occurred (whatever that means). |

| | |
|---|---|
| 5 | Error reading a cluster. |
| 6 | Error writing a cluster. |
| 7 | An allocation error occurred. |
| 8 | Memory error. |
| 9 | Insufficient memory. |

## Examples

Use the following command to execute DEFRAG in interactive mode:

```
defrag
```

Use the following command to perform a full optimization on drive C, sorting files with the newest files first and forcing a reboot on completion.

```
defrag c: /f /s:-d /b
```

CROSS REFERENCE

## Additional information

Refer to Chapter 4 for a thorough discussion of disk defragmentation.

*See also* CHKDSK and DBLSPACE /DEF.

# DEL (ERASE)

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 6 | ■ | ☒ | ☒ | ☐ |

## Description

This command deletes one or more files.

## Usage

You can execute the command as either DEL or ERASE:

```
del [drive:][path]wildcard.ext [/p]

erase [drive:][path]wildcard.ext [/p]
```

## Options and switches

[*drive:*][*path*]*wildcard.ext*

This option specifies the name of the file or files to be deleted. If a filename is not specified and a drive or path is specified, DOS assumes the wildcard *.*.

```
[/p]
```

This switch instructs DOS to prompt you individually before deleting each file. Even if the /P switch is omitted, DOS prompts for confirmation before deleting all the files in a directory.

## Remarks

The information stored in the file is not erased. Behind the scenes, the filename is marked with a special character — ASCII character 229, to be precise, called *sigma.* The clusters the file occupies are marked as available; when a new file is created, these clusters can be used to store the data for the new file. That the file's data is not erased immediately makes file undeletion possible, even when no backup has been made.

A shortcut for deleting every file in the current directory is to use a single period (.) rather than *.*.

If you try to delete a file that has the read-only attribute set, DOS displays a message that access is denied. You can always use the ATTRIB command to remove the read-only flag, but be careful. The file may be marked as read-only for the express purpose of preventing you from deleting it.

## Examples

You can use any of the three following commands to delete every file in the TRASH directory:

```
del \trash

del \trash\.

del \trash\*.*
```

Use the following command to delete all BAK files in the current directory:

```
del *.bak
```

To instruct DOS to prompt you before deleting every WK1 file in the 123 directory, enter the following command:

```
del c:\123\*.wk1 /p
```

## Additional information

CROSS REFERENCE

Refer to Chapter 3 for more information about precautions and measures you can take to avoid accidental data loss.

*See also* MSBACKUP, UNDELETE, and DELTREE.

### Bob's Shareware Collection disks

CMFiler, from NoVaSoft, is a full-screen file manager that provides, in addition to many other things, easy ways to tag and delete files.

To delete a classified or sensitive file and ensure that no one (and no program) can undelete it, delete it with RELDEL, from TARDIS DP Consultants. RELDEL overwrites the file's data, shrinks the file to zero bytes, and then deletes it.

## DELOLDOS

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☐ | ☐ |

### Description

When you install DOS 6, a copy of your old version of DOS is saved in a special subdirectory. Use DELOLDOS to delete the copy of the old DOS version and free up disk space.

### Usage

```
deloldos [/b]
```

### Options and switches

```
[/b]
```

This switch instructs DELOLDOS to use monochrome colors. Alternatively, you can press F5 to switch to monochrome while DELOLDOS is running.

### Remarks

When you install DOS 6, SETUP makes a backup copy of your old DOS files in a directory named OLD_DOS.1. What's more, if you install DOS 6 more than once, directories OLD_DOS.2 and so on are created. This facility is most valuable to beta testers who install DOS 6 a number of times — as a general user, you normally load DOS 6 only one time.

The OLD_DOS.*n* directory contains all the files from the old DOS directory, along with copies of the system files from the boot directory and copies of AUTOEXEC.BAT and CONFIG.SYS (renamed with a DAT extension).

The uninstall disk (created by SETUP) includes the program UNINSTALL, which deletes the DOS 6 files and restores the old copy of DOS by transferring files from the OLD_DOS.*n* directory to their original locations.

You don't need to keep the OLD_DOS.*n* directory for more than a few weeks while you become confident that DOS 6 is stable on your system. After a few weeks of trouble-free DOS 6 use, run DELOLDOS to remove the old DOS files.

### Example

To delete the copy of your old DOS version, just enter the following command:

```
deloldos
```

CROSS REFERENCE

### Additional information

*See also* UNINSTALL.

CAUTION

If you run DELOLDOS, you cannot run UNINSTALL to restore your old version of DOS. Don't run DELOLDOS until a few weeks after installing DOS 6, when you are confident that you will not need to UNINSTALL it.

TIP

If you compress your boot drive with DBLSPACE, you cannot run the UNINSTALL program, and the old version of DOS becomes useless. If you DBLSPACE your drive, you might as well run DELOLDOS to create even more disk space.

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

## DELTREE

DOS 6

### Description

This powerful and potentially destructive command removes a directory, all of its subdirectories, and all the files in the directories — just like that.

### Usage

deltree [/y] [*drive:*]*path*

### Options and switches

[*drive:*]*path*

This option is the name of the directory to be deleted. This directory and all of its subdirectories are removed. You can specify wildcards in the directory name, but you must be careful — if a wildcard matches a filename, the file also is deleted.

```
[/y]
```

This switch suppresses the warning prompt (deletes the files and directories with prompting for confirmation).

## Remarks

DELTREE returns a non-0 ERRORLEVEL if the command was aborted or 0 ERRORLEVEL if the specified tree was deleted.

## Examples

Use the following command to remove the \TRASH directory and all its files and subdirectories:

```
deltree \trash
```

To delete the \123W\SAMPLE directory and bypass the confirmation prompt, enter the following command:

```
deltree /y \123w\sample
```

To delete all directories beginning with *TR* and their subdirectories, enter the following command:

```
deltree \tr*.
```

## Additional information

See Chapter 1 for more information on DELTREE.

*See also* DEL and RD.

## Bob's Shareware Collection disks

The Bob's Shareware Collection disks are packed with tools to aid file deletion chores. CMFiler, the file manager from NoVaSoft, provides visual tools for deleting entire directory trees — I prefer it because I can easily see what is being deleted. PDELETE, from Patri-Soft, and PocketD, from Pocketware, both provide extensive and flexible file deletion options using command-line switches.

Although this command is useful, it is also very powerful. DELTREE will delete all files contained in a directory and its subdirectories regardless of attributes. With one brief command, you can erase all the files on your hard disk. Never use this command while you are distracted — think before you use DELTREE!

# DEVICE

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

## Description

This CONFIG.SYS statement loads a device driver into conventional or base memory.

## Usage

```
device=[drive:][path]filename.ext [device-parameters]
```

## Options and switches

You can use the device statement only in the CONFIG.SYS file.

[*drive:*][*path*]*filename.ext*

This option is the name of the device driver file. If the file is not located in the root directory of the boot disk, you should specify the full path.

[*device-parameters*]

This option specifies the additional parameters and switches that some device drivers must have when they are loaded.

## Remarks

Device driver software is used to provide software links to a hardware device such as a scanner, mouse, or fax card. The device drivers are supplied with hardware and must be installed in CONFIG.SYS before the hardware can be used. DOS 6 includes a variety of device drivers for managing memory, the display, and connected computers.

CONFIG.SYS often includes multiple device statements, and the order can be important. HIMEM.SYS, for example, must be installed before EMM386.EXE. Vendors indicate whenever their device drivers must be loaded in a special order.

Most, but not all, device drivers have a SYS extension. Some programs have dual purposes and can be loaded as devices and executed from the command line. DOS 6's SMARTDRV.EXE is an example.

### Example

The following example shows a plain CONFIG.SYS that includes three DEVICE statements:

```
FILES=20
BUFFERS=15
DEVICE=C:\FCONSOLE.DEV
DEVICE=C:\WINDOWS\MOUSE.SYS /Y
DEVICE=C:\DOS\SMARTDRV.EXE
LASTDRIVE=P
```

### Additional information

Device drivers installed with DEVICE are normally loaded into base (or conventional) memory. You can use the DEVICEHIGH command to load devices into UMBs.

CROSS REFERENCE

See Chapter 5 for more information on DEVICE.

*See also* DEVICEHIGH, INSTALL, and MEMMAKER.

## DEVICEHIGH

BOB'S SCORECARD 9

PROMPT ☐ AUTOEXEC ☐ BATCH ☐ CONFIG ■

### Description

Gotta have it. DEVICEHIGH, the sister to DEVICE, is used to load device drivers into upper memory.

### Usage

```
devicehigh=[drive:][path]filename.ext [device-parameters]
```

### Options and switches

The DEVICE statement can be used only in CONFIG.SYS.

[*drive:*][*path*]*filename.ext*

This option specifies the name of the device driver file. If the file is not located in the root directory of the boot disk, you should specify the full path.

[*device parameters*]

This option specifies the additional parameters and switches required by the device driver.

***Note:*** See the “Extended use” section for additional switches.

## Remarks

Before device drivers can be installed with DEVICEHIGH, the upper memory area must be configured using the HIMEM.SYS, EMM386.EXE, and DOS=UMB statements.

The order of the DEVICEHIGH statements in CONFIG.SYS has a real impact on the amount of free memory. Tuning the DEVICEHIGH statements to optimize memory use can be a lengthy and time-consuming process. The MEMMAKER command, however, can completely automate this difficult task.

If insufficient memory exists to load a device into upper memory, DEVICEHIGH uses base memory, just as DEVICE does.

You can load programs and device drivers into upper memory by using the LOADHIGH command.

Refer to the DEVICE command for additional remarks regarding device drivers.

## Example

The following example shows a simple CONFIG.SYS file that uses DEVICEHIGH to load the MOUSE.SYS device driver into upper memory:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM I=B000-B7FF NOEMS
BUFFERS=15,0
FILES=50
DOS=HIGH,UMB
DEVICEHIGH=C:\WINDOWS\MOUSE.SYS /Y
```

## Additional information

Refer to Chapter 5 for a thorough discussion of memory management techniques.

*See also* DEVICE, EMM386.SYS, HIMEM.SYS, LOADHIGH, and MEMMAKER.

## Bob’s Shareware Collection disks

DOS 6 supports DEVICEHIGH on only 386 and later systems. UMB_DRV.SYS, Christopher Blum’s excellent memory manager, provides advanced memory capabilities to 286 users as well as users of 386 and later systems.

### Extended use

Although DEVICEHIGH loads the device driver into the largest free area of upper memory by default, this step is not always efficient. To provide more control over memory use, DEVICEHIGH also supports the /L and /S switches. You normally should leave the responsibility of determining the best settings for these switches to MEMMAKER.

The following example shows the syntax for these two advanced switches:

```
[[/l:region][,minsize][;region2][,minsize2]]
```

The /L switch is used to control the region of UMBs in which the device is loaded. (The MEM /F command displays the free regions of upper memory. Device drivers can be too large to fit in one region.) The /L switch enables you to specify a second region. You can use the MINSIZE parameter to ensure that adequate memory is allocated for the device — some device drivers require more memory during operation than they do when they are being initialized.

Thanks to MEMMAKER, the following example configuration uses the /L switch:

```
DEVICEHIGH /L:2,12048 =C:\DOS\SETVER.EXE
DEVICEHIGH /L:2,55168 =C:\WINDOWS\MOUSE.SYS /Y
DEVICE=C:\U\FAX\SATISFAX.SYS IOADDR=0350
DEVICEHIGH /L:2,11712 =C:\PROAUDIO\MVSOUND.SYS D:5 Q:7 J:0
DEVICEHIGH /L:2,33696 =C:\PROAUDIO\TSLCDR.SYS /D:MVCD001 /R /P:3
```

```
[/s]
```

The /S switch (for *shrink*) instructs DEVICEHIGH to reduce the allocated memory after the device driver has been initialized. This switch is used with device drivers that need more memory to load than they do during subsequent operation.

## DIR

**BOB'S SCORECARD** 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

This command displays information about one or more files, including name, size, and the date and time of last creation.

### Usage

```
dir [drive:][path][wildcard.ext] [/p] [/w] [/a[:]attr] [/o[:]order]
[/s] [/b] [/l] [/c]
```

## Options and switches

When you use DIR with no switches, the switches specified in the environment variable DIRCMD are used. If DIRCMD is not set, all files, excluding hidden files, are displayed in a single column.

`[`*drive:*`][`*path*`][`*wildcard.ext*`]`

This option specifies the file or files whose details are listed. If no filemask is specified, a wildcard of *.* is assumed.

`[/p]`

This switch instructs DOS to *p*ause for a keypress after filling the display with file details.

`[/w]`

This switch instructs DOS to display the filenames in *w*ide format with five filenames to a row. No file details are displayed when this switch is used.

`[/a[:]`*attr*`]`

This switch displays only files that match the specified attributes. The following codes are used with the /A switch to identify the attributes:

| ***Code*** | ***Attribute*** |
|---|---|
| H | List hidden files |
| –H | List files that aren't hidden |
| S | List system files |
| –S | List nonsystem files |
| D | List directories but not files |
| –D | List files but not directories |
| A | List files with the archive attribute set |
| –A | List files that don't have the archive attribute set |
| R | List read-only files |
| –R | List files that are not read-only |

You can specify multiple codes, but do not separate them with spaces (/A:HS–R, for example). If you specify the /A switch with no codes, all files are listed. If you omit the entire switch, DIR lists all files in the current directory except hidden and system files.

The colon is optional but should be used for clarity.

```
[/o[:]order]
```

This switch identifies the order in which files are displayed (sorted, for example). The following codes are used with the /O switch to control the sort order:

| *Code* | *Sort Order* |
|---|---|
| N | Sorted by filename in ascending order (A first) |
| –N | Sorted by filename in descending order (Z first) |
| E | Sorted by extension in ascending order |
| –E | Sorted by extension in descending order |
| D | Sorted in date/time order (oldest files first) |
| –D | Sorted in date/time order (most recent files first) |
| S | Sorted by file size (smallest first) |
| –S | Sorted by file size (largest first) |
| G | Grouped with directories displayed before files |
| –G | Grouped with files displayed before directories |
| C | Sorted by DBLSPACE compression ratio (least compressed first) |
| –C | Sorted by DBLSPACE compression ratio (most compressed first) |

You can specify multiple codes, but do not separate them with spaces. If you specify the /O switch with no codes, the switch /O:GN is applied (sorted in ascending filename order with directories grouped first). If you omit the entire switch, DIR lists the files in the unsorted (or FAT table) order.

The colon (:) in the /O command is optional but should be used for clarity.

```
[/s]
```

The /S DOS switch is my favorite switch of all time. It instructs DOS to list the files in the specified directory *and* all subdirectories. It provides a convenient (if not lightning-fast) way to search for a file or files across multiple directories.

```
[/b]
```

This switch displays the filenames in a brief format, with only the filenames and no file details, and suppresses the standard heading and summary information. If you specify both /B and /W, /B is used, and /W is ignored.

```
[/l]
```

By default, all filenames and details are displayed in uppercase. The /L instructs DIR to display all alphabetical characters in lowercase.

```
[/c]
```

This switch displays DBLSPACE file-compression ratios for every file. This switch is ignored when /B or /W also is specified.

## Remarks

If you regularly use one or more switches with the DIR command, you can save yourself some keystrokes by using the DIRCMD environment variable. DIR applies any switches associated with the DIRCMD environment variable every time it is executed. To always sort the file list in alphabetical order with directories first and the pause switch in effect, for example, enter the following command:

```
set dircmd=/o:gne/p
```

DIR applies those switches thereafter and any others you specify on the command line. You might consider adding the SET statement to AUTOEXEC.BAT, to apply the defaults automatically. To disable an environment switch temporarily, enter the switch preceded by a minus sign. Having executed the preceding command to set DIRCMD, you can execute DIR without the pause switch by entering the following command:

```
dir /-p
```

To clear the DIRCMD settings, enter the following command:

```
set dircmd=
```

CROSS REFERENCE

You can simplify the DIR command by using DOSKEY macros. Refer to Chapter 9 for more information.

## Examples

Use the following command to display all the files with a DOC extension in the WORD directory and pause the output after every full screen:

```
dir \word\*.doc /p
```

Use the following command to display all hidden files on drive C:

```
dir c:\ /a:h /s
```

To display on drive D all the directories sorted in alphabetical order, use the following command:

```
dir d:\ /a:d /o:n /b /s
```

The following command displays all the files in the current directory sorted by name and then by extension, using lowercase characters:

```
dir /o:ne /l
```

## Additional information

*See also* ATTRIB, DBLSPACE, DOSKEY, SET, and TREE.

## Bob's Shareware Collection disks

Robert Woeger's elegant utility HotDIR Plus provides additional capabilities for listing files. The most obvious is the use of colors to differentiate between different file types and directories. (You can even configure these colors.) HOTDIR PLUS also has a six-column display option.

PocketD provides excellent file listing capabilities, including full color support, powerful sorting and selection options, flexible filemask support, and a special Tiny Mode which can show hundreds of files in a single screen.

If you are using DIR to search for files, try Bruce Gavin's superb QFIND program. It is blazingly fast and has a wealth of features, including searching inside zip files, searching multiple drives, optional directory change when a match is found, and much more.

If you cannot remember where a file is located and you don't have a file-finder utility, just use the DIR command with the /S switch. To find the file SECRET.DOC on drive C, for example, enter the following command:

```
dir c:\secret.doc /s
```

This secret shouldn't be a secret, but few people realize that the /A switch (with no codes) displays all files including hidden and system files. Use the following command to display all files in the root directory of drive C:

```
dir c:\ /a
```

BOB'S SCORECARD 1

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

# DISKCOMP

## Description

This command compares two floppy disks to see whether they contain identical data.

## Usage

`diskcomp` *drive1: drive2:* `[/1] [/8]`

## Options and switches

*drive1: drive2:*

This option specifies the drive letters that identify the drives containing the disks to be compared. If you have only one drive to compare two disks, use the same letter twice (`a: a:`, for example), but get ready for some serious floppy swapping.

`[/1]`

This switch instructs DISKCOPY to compare only the first side. Nowadays, all floppies use both the upper and lower surfaces of the disk; in the early days of the PC, however, single-sided floppies were common.

`[/8]`

This switch, inherited from the days of DOS 2.0, is used to compare only the first eight tracks.

## Remarks

DISKCOMP compares the disks on a track-by-track basis. Even if the two disks contain the identical files, DISKCOMP reports a discrepancy if the files are stored physically in a different order.

DISKCOMP does not compare two hard drives, nor does it compare floppy disks of different media types.

When you use DISKCOMP in a batch file, the following ERRORLEVELs may be returned:

| ***ERRORLEVEL*** | ***Message*** |
|---|---|
| 0 | The disks are identical. |
| 1 | The disks are not identical. |
| 2 | The user aborted the comparison (Ctrl-Break was pressed). |
| 3 | DISKCOMP could not read one of the disks. |
| 4 | An unexpected internal error occurred. |

I rarely use this cantankerous old command, but it's there if you need it.

## Examples

Use the following command to compare the disks in drives A and B:

```
diskcomp a: b:
```

To compare two disks when you only have one floppy disk drive, enter the following command:

```
diskcomp a: a:
```

## Additional information

Use the FC command to compare two files.

CROSS REFERENCE

*See also* DISKCOPY.

# DISKCOPY

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

## Description

This command duplicates all the data from one floppy disk to another.

## Usage

```
diskcopy [sourcedrive:] [targetdrive:] [/1] [/v]
```

## Options and switches

`[`*sourcedrive:*`]` `[`*targetdrive:*`]`

This option specifies the source and target drives. You can specify the same drive for both drives, and DOS prompts you (many times) to switch disks during the copy operation. If you omit the target drive, the current drive is assumed. If you omit both drives, the current drive is used for the source and the target.

`[/1]`

This old switch instructs DISKCOPY to copy only the first side; because all disks are now double sided, however, you probably will never use this switch.

`[/v]`

This switch instructs DISKCOPY to perform a CRC check to *v*erify that the data written to the target can be read successfully. It does not guarantee that the source and target disks are identical — use DISKCOMP for true verification.

## Remarks

You can use DISKCOPY to duplicate only floppy disks, not hard disks.

When you use DISKCOPY, any data on the target disk is overwritten.

If the target disk is not formatted, DISKCOPY automatically formats it during the copy procedure.

Because DISKCOPY copies the data on a track-by-track basis, the disk media must be identical. You cannot use DISKCOPY to duplicate, for example, a 1.2MB 5¼-inch disk to a 1.44MB 3½-inch disk. Use XCOPY to copy data on a file-by-file basis from one media type to another.

You can use DISKCOPY in a batch file, and it returns the following ERRORLEVELs to indicate the completion status:

| *ERRORLEVEL* | *Message* |
|---|---|
| 0 | The copy operation was successful. |
| 1 | An error occurred in reading or writing data. |
| 2 | The user aborted the copy (Ctrl-Break was pressed). |
| 3 | A critical (hardware) error occurred. |
| 4 | An unexpected internal error occurred. |

## Examples

Enter the following command to duplicate the disk in drive A to a target disk in drive B:

```
diskcopy a: b:
```

Use the following command to duplicate a disk by using the single drive A:

```
diskcopy a: a:
```

### Additional information

*See also* DISKCOMP and XCOPY.

### Bob's Shareware Collection disks

DUPE, from Best Technology, provides a fast and convenient way to duplicate disks, without all the floppy swapping.

## DISPLAY.SYS

BOB'S SCORECARD 3

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This command provides international character support by allowing code-page switching.

### Usage

The DISPLAY.SYS device driver must be loaded in CONFIG.SYS with either a DEVICE= or DEVICEHIGH= statement as follows:

```
device[high]=[drive:][path]display.sys con[:]=(type[,[hwcp][,n]])
```

### Options and switches

[*drive:*][*path*]

This option identifies the directory in which DISPLAY.SYS is located, if it is not in the boot directory. SETUP places DISPLAY.SYS in the \DOS directory by default.

*type*

This option identifies the type of installed display adapter. Specify EGA for EGA and VGA systems, or LCD for LCD displays. The types CGA and MONO are allowed also, but because these devices don't support code-page switching, DISPLAY.SYS serves no purpose on such systems.

[*hwcp*]

This option identifies that *h*ard*w*are *c*ode *p*age, and can have one of the following values:

| ***Code*** | ***Value*** |
|---|---|
| 437 | United States |
| 850 | Multilingual (known as Latin page I) |
| 852 | Slavic (known as Latin page II) |
| 860 | Portuguese |
| 863 | Canadian-French |
| 865 | Nordic |

[*n*]

This option informs DISPLAY.SYS of the number of additional code pages supported by the device. The value is hardware dependent; as a rule of thumb, however, use 6 for EGA/VGA, 1 for LCD, and 0 for CGA and MONO.

## Remarks

You should use this device driver only if you are configuring your system for international use.

## Example

Use the following statement in CONFIG.SYS to configure a VGA system for Portugal with one additional code page:

```
device=c:\dos\display.sys con=(ega,860,1)
```

## Additional information

Refer to Chapter 9 of the *MS-DOS User's Guide* for further information about configuring DOS for international use.

*See also* CHCP, DEVICE, DEVICEHIGH, KEYB, MODE, NLSFUNC, and PRINTER.SYS.

## Extended use

DISPLAY.SYS optionally supports code-page subfonts with the following syntax:

```
device[high]=[drive:][path]display.sys con[:]=(type[,[hwcp][,(n,m)]])
```

where *m* specifies the number of subfonts the display hardware provides. Subfonts are normally used for storing different-size characters (the characters used in 43-line mode, for example). The default values are 2 for EGA, 1 for LCD, and 0 for CGA and MONO.

# DOS

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

## Description

Controls whether DOS is loaded in high memory (on 286 systems and higher) and whether DOS should manage the upper memory area (on 386 systems and higher).

## Usage

The DOS statement is used in CONFIG.SYS. Two DOS= entries are supported using the following syntax:

```
dos=high or low
dos=umb or noumb
```

Alternatively, you can use one of the following formats to combine the HIGH and UMB identifiers in a single DOS= statement:

```
dos=[high or low][,umb or noumb]
```

## Options and switches

```
[high or low]
```

Use the HIGH switch to instruct DOS to load some of its code in high memory (HMA) or LOW to load the entire operating system in base memory. Normally, you do not specify LOW because it is the default.

```
[umb or noumb]
```

Use the UMB switch to instruct DOS to manage the upper memory area or NOUMB to instruct DOS to ignore UMBs. Again, you do not normally specify NOUMB because it is the default.

## Remarks

You must load the extended memory manager HIMEM.SYS, or a third-party equivalent, before the DOS=HIGH instruction can be installed. DOS cannot be loaded HIGH on 8086 systems.

Similarly, you must load EMM386.EXE, or a third-party equivalent, and configure it to provide UMBs before the DOS=UMB statement can be processed. UMBs are supported on only 386 or later systems.

### Examples

The following CONFIG.SYS file loads DOS into high memory and instructs DOS to manage UMBs:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE RAM
DOS=HIGH,UMB
DEVICEHIGH=C:\WINDOWS\MOUSE.SYS /Y
```

On a 286 system, the CONFIG.SYS has to be adjusted to remove the UMB support as follows:

```
DEVICE=C:\DOS\HIMEM.SYS
DOS=HIGH
DEVICE=C:\WINDOWS\MOUSE.SYS /Y
```

CROSS REFERENCE

### Additional information

Refer to Chapter 5 for a thorough discussion of memory management.

*See also* DEVICEHIGH, EMM386.EXE, HIMEM.SYS, LOADHIGH, MEM, MEMMAKER, and MSD.

DISK

### Bob's Shareware Collection disks

UMB_DRV.SYS, from Christopher Blum, provides advanced memory management capabilities for all PC types including 8086 and 80286 systems.

TIP

On 386 systems and higher, MEMMAKER.EXE can update CONFIG.SYS automatically and, if necessary, add HIMEM.SYS, EMM386.EXE, and DOS=UMB statements. It doesn't, however, add a DOS=HIGH statement because SETUP adds it when DOS 6 is installed.

## DOSHELP

*See* FASTHELP.

## DOSKEY

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ■ | ☒ | ☐ |

### Description

DOSKEY provides three primary and very useful functions: the ability to recall previous commands to save retyping, command-line editing (no more backspace), and a simple but effective macro utility.

### Usage

```
doskey [/insert or /overstrike] [/reinstall] [/bufsize=size]
[/macros] [/history] [macroname=[text]]
```

### Options and switches

By running DOSKEY with no switches, you gain command recall and command editing.

```
[/insert or /overstrike]
```

Just like a standard editor, DOSKEY supports insert and overstrike modes for editing the command line. This switch is used to control whether DOSKEY starts in overstrike or insert mode. If you do not specify this switch, DOSKEY starts in overstrike mode. While you are editing, you can switch or toggle the mode by pressing Ins.

```
[/reinstall]
```

The first time you execute DOSKEY in a session, the program is loaded in memory as a TSR and records every DOS command as it is executed. To clear the history of old commands, you can reexecute DOSKEY with the /REINSTALL switch. You must use the switch also if you want to reinstall DOSKEY with a larger buffer.

```
[/bufsize=size]
```

DOSKEY uses a buffer to store the history of executed commands and DOSKEY macros. By default, 512 bytes of memory are reserved for this buffer. You can use the /BUFSIZE switch to allocate a different buffer size, but 256 is the minimum.

```
[/macros]
```

This switch displays a list of all active DOSKEY macros. The /M switch has the same meaning.

```
[/history]
```

This switch displays a list of all the previous DOS commands stored in DOSKEY's buffer. The /H switch has the same meaning.

[*macroname*=[*text*]]

This option defines a macro and assigns it a specific name. Thereafter, whenever the user enters the *macroname* on the command line, DOSKEY executes the *text* associated with the macro. It provides a great way to customize DOS and simplify complex commands. You can delete a macro by specifying MACRONAME= with no *text*.

## Remarks

Having installed DOSKEY, you can recall previous commands by pressing the up- and down-arrow keys. PgUp recalls the oldest command, and PgDn the most recent.

The following table shows the keys you can use to edit the command line.

| *Key* | *Action* |
|---|---|
| Left arrow | Move left one character |
| Right arrow | Move right one character |
| Ctrl-left arrow | Move left one word |
| Ctrl-right arrow | Move right one word |
| Home | Move to the beginning of the command line |
| End | Move to the end of the command line |
| Esc | Erase the command |

If you don't use a third-party alternative, I recommend that you use DOSKEY all the time by adding it to your AUTOEXEC.BAT file. DOSKEY is a TSR and on 386 systems and higher, you can save base memory by installing DOSKEY with LOADHIGH.

## Examples

Use the following line from AUTOEXEC.BAT to load DOSKEY into high memory and set it to Insert Mode:

```
loadhigh c:\dos\doskey /insert
```

To create a macro, BAKFIG, for making a backup of CONFIG.SYS, enter the following command:

```
doskey bakfig=copy c:\config.sys c:\config.bak
```

To reload DOSKEY with a buffer size of 800 bytes and clear the command history, enter the following command:

```
doskey /reinstall /bufsize=800
```

### Additional information

Refer to Chapter 9 for a thorough explanation of DOSKEY.

*See also* LOADHIGH.

### Bob's Shareware Collection disks

Moderne Software's excellent command-line utility, Anarkey, offers all the features of DOSKEY, and then some. For example, you can load and save command histories to disk.

A little-known feature of DOSKEY is that it enables you to execute multiple commands at one time, much like OS/2 does. Having typed the first command, press Ctrl-T and then type the next command, press Ctrl-T again and enter the third command, and so on. DOS displays the paragraph character (¶) to show you where Ctrl-T was pressed. When you press Enter, every command is executed in turn.

To make the directory DEMO and then change to the directory, for example, you enter the following command:

```
c:\>md demo ¶ cd demo
```

BOB'S SCORECARD 5

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **DOSSHELL** | ■ | ☒ | ☒ | ☐ |

## DOSSHELL

### Description

This command starts the DOS Shell program. DOS Shell provides menuing, file-management, and task-switching tools.

### Usage

```
dosshell [/t[:res]] [/b]
```

or

```
dosshell [/g[:res]] [/b]
```

## Options and switches

The DOSSHELL switches are used only to control the display characteristics. If you run DOSSHELL with no switches, the settings from the last DOSSHELL session are used.

```
[/t]
```

This switch starts the DOSSHELL in Text Mode.

```
[/g]
```

The /G switch starts the DOSSHELL in Graphics Mode.

```
[:res]
```

This option indicates which resolution to use and, optionally, the number of lines. The resolution is coded with one of three letters: *l* for low resolution, *m* for medium resolution, and *h* for high resolution. The supported resolutions are hardware dependent; the following table shows the possible permutations:

| ***Switches*** | | ***Number of Display Lines*** | | |
|---|---|---|---|---|
| ***Mode*** | ***Resolution*** | ***Mono/CGA*** | ***EGA*** | ***VGA*** |
| /T: | l | 25 | 25 | 25 |
| /T: | m,m1,h,h1 | N/A | 43 | 43 |
| /T: | m2,h2 | N/A | 43 | 50 |
| /G: | l | 25 | 25 | 25 |
| /G: | m,m1 | N/A | 43 | 30 |
| /G: | m2 | N/A | 43 | 34 |
| /G: | h,h1 | N/A | 43 | 43 |
| /G: | h2 | N/A | 43 | 60 |

```
[/b]
```

This switch instructs DOSSHELL to use monochrome colors.

## Remarks

If you cannot be bothered by all these switches, just run DOSSHELL with no switches and choose the display settings you want from the Options Display menu.

The DOSSHELL.INI file contains the detailed initialization settings for DOSSHELL.

If you use Windows, DOSSHELL has limited value; File Manager and Program Manager are clearly superior. Don't load Windows from within the DOSSHELL — quit to DOS first.

### Examples

Use the following command to start the DOSSHELL in black-and-white mode:

```
dosshell /b
```

Use the following command to start the DOSSHELL in 34-line graphics mode on a VGA system:

```
dosshell /g:m2
```

### Additional information

CROSS REFERENCE

Chapter 2 has further information about using and customizing the DOS Shell.

BOB'S SCORECARD 4

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

## DRIVER.SYS

### Description

This device driver creates logical floppy drives with different characteristics from the physical drive. You can use it also to extend the BIOS on old systems to enable newer drive types to be installed.

### Usage

```
device[high]=[drive:][path]driver.sys /d:number [/c] [/f:factor]
```

### Options and switches

`[drive:][path]`

This option identifies the directory in which DRIVER.SYS is located, if it is not in the boot directory. SETUP places DRIVER.SYS in the \DOS directory by default.

`/d:number`

This switch identifies the drive number of the physical *floppy* drive. Drive A has a value of 0, if present, drive B has a value of 1, and so on.

`[/c]`

Most modern drives have change-line support. That is, they can sense quickly when the drive door is open. If your physical drive supports change-line support (most do), specify /C to speed up floppy disk access.

`[/f:`*factor*`]`

This switch identifies the disk drive type. One of the following values is accepted for the *factor*:

| *Value* | *Drive Type* |
|---|---|
| 0 | 360K or less 5¼-inch drive |
| 1 | 1.2MB 5¼-inch drive |
| 2 | 720K 3½-inch drive |
| 3 | 1.44MB 3½-inch drive |
| 4 | 2.88MB 3½-inch drive |

If you do not specify this switch, DOS assumes /F:2.

## Remarks

DRIVER.SYS is not one of your everyday commands. Its use has diminished with the dominance of 1.2 and 1.44 drives. You should use it only if your system BIOS does not recognize the drive type or if you are adding an external drive (installing a 1.44 drive on an original IBM PC-AT, for example).

You might consider using DRIVER.SYS to create a logical drive that has different characteristics from the physical drive. If you have a 1.2MB drive, for example, you might configure an additional *logical* drive D to be a 360K floppy drive. When you access drive D, you are using drive A, but DOS treats it as though it were a 360K drive. This technique saves you from having to remember FORMAT switches and the like.

The drive letter assigned to the logical drive depends on the number of drives already assigned. At the time the DRIVER.SYS statement is processed, the next available drive letter is used.

## Examples

Use the following CONFIG.SYS entry to support an external 1.44MB floppy drive:

```
device=c:\dos\driver.sys /d:2 /f:7
```

Use the following command to create a logical drive D (assuming that you have only one hard drive), which uses drive A as though it were a 360K drive:

```
device=c:\dos\driver.sys /d:0 /f:0
```

### Additional information

CROSS REFERENCE

*See also* DEVICE, DEVICEHIGH, DRIVPARM, and FORMAT.

### Extended use

If the physical drive does not match any of the /F:FACTOR characteristics, you can use the following three switches to define the custom drive type:

```
[/h:heads]
```

This switch specifies the number of read-write drive heads. The default is 2, but values in the range from 1 to 99 are accepted.

```
[/s:sectors]
```

This switch specifies the number of sectors per track. Values in the range from 1 to 99 are accepted.

```
[/t:tracks]
```

This switch specifies the number of tracks per drive. Values in the range from 1 to 999 are accepted.

## DRIVPARM

BOB'S SCORECARD 2

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This command is used to define or redefine the properties of disk and tape drives (block devices).

### Usage

You should place the DRIVPARM command in CONFIG.SYS.

```
drivparm=/d:number [/c] [/f:factor] [/h:heads] [/i] [/n]
[/s:sectors] [/t:tracks]
```

## Options and switches

`/d:`*number*

This switch identifies the physical device number of the drive. Drive A is 0, drive B is 1, drive C is 2, and so on. The value must be in the range from 0 to 255.

`[/c]`

Use this switch when the device can detect that the drive door is open (like DRIVER.SYS).

`[/f:`*factor*`]`

This switch identifies the drive type, where *factor* can have one of the following values:

| *Value* | *Drive Type* |
|---|---|
| 0 | 360K or less 5¼-inch drive |
| 1 | 1.2MB 5¼-inch drive |
| 2 | 720K 3½-inch drive |
| 3 | (No longer supported) |
| 4 | (No longer supported) |
| 5 | Hard disk |
| 6 | Tape drive |
| 7 | 1.44MB 3½-inch drive |
| 8 | Read/write optical drive |
| 9 | 2.88MB 3½-inch drive |

If you do not specify this switch, DRIVPARM uses /S:2.

`[/h:`*heads*`]`

This switch identifies the number of read/write heads on the drive. The default depends on the drive type, but values in the range from 1 to 99 are accepted.

`[/i]`

This switch indicates that the drive is a 3½-inch floppy disk. You should use it when the ROM BIOS doesn't support 3½-inch drives.

`[/n]`

The /N switch indicates that the drive contains nonremovable media (a hard disk, for example).

`[/s:`*sectors*`]`

This switch identifies the number of sectors per track that the drive supports. The default depends on the drive type, but values in the range from 1 to 99 are accepted.

`[/t:`*tracks*`]`

This switch identifies the number of tracks per side that the drive supports. The default depends on the drive type, but values in the range from 1 to 99 are accepted.

### Remarks

DRIVPARM is a highly specialized command you normally use only when instructed to do so by a hardware vendor.

Unlike DRIVER.SYS, DRIVPARM can modify only the properties of an existing drive and cannot be used to create additional logical drives.

If the affected drive is installed through a device driver, the DRIVPARM statement should be located in CONFIG.SYS on the line immediately after the drive's primary device driver.

### Additional information

*See also* DRIVER.SYS.

## ECHO

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ■ | ■ | ☐ |

### Description

This command controls how batch file commands are displayed during execution.

### Usage

You can use the ECHO command in one of the following two forms:

```
echo [on or off]
```

```
echo [message]
```

## Options and switches

When you use ECHO without switches, it displays the current echoing status (whether echoing is on or off). This feature is not one I use every day.

`[on or off]`

DOS displays every command line of a batch file by default as the line is executed. If it encounters the ECHO OFF command, DOS suppresses the command-line displays (or echoes). You can turn command echoing on again with the command ECHO ON.

`[message]`

When echoing is turned off, you can use the message switch to display a message.

## Remarks

Batch files are often used to simplify complex or repetitive tasks. The ECHO statement is designed to reduce the screen clutter that occurs when these batch files are executed. Think of it as a *clean screen* setting.

Every time a batch file executes, the ECHO status defaults to on.

You can suppress the echoing of *any* single command individually by prefixing the command with the @ character. Standard practice is to start a batch file with the command @ECHO OFF, to suppress the echoing of the first statement and all subsequent statements.

## Examples

The following batch file displays a message and waits for user input:

```
@echo off
CHOICE Do you want to load the mouse
if errorlevel 2 goto quit
mouse /y
:quit
```

The following batch file clears the screen and uses ECHO to display a message:

```
@echo off
cls
echo Searching for hidden files....
dir \*.* /a:h /s
echo Search complete
```

## Additional information

Refer to Chapter 10 for more information about batch file programming techniques.

*See also* ANSI.SYS, PAUSE, and CHOICE.

## Bob's Shareware Collection disks

BATUTIL, from CtrlAlt Associates, provides some excellent tools for sprucing up batch file output.

ECHO suppresses the display of the line to be executed, but it does not suppress the output generated by the executing command. You can often suppress unwanted output from command-line utilities by using `>nul` to pipe the output to a nul device. The following batch file, for example, copies CONFIG.SYS but suppresses the output from the COPY command:

```
@echo off
echo Backing up system files
del c:\config.bak > nul
copy c:\config.sys c:\config.bak > nul
if exist c:\config.sys echo Copy successful!
```

To display a blank line (for cosmetic reasons), don't use just the command ECHO with no message. Because DOS assumes that you want to determine the active echo setting, it displays a message that ECHO is on or ECHO is off — not very useful!

To display a blank line, use the ECHO command followed immediately by a period (.) with no spaces. The following batch file, for example, displays a menu item on every other line:

```
@echo off
cls
echo 1) Lotus 1-2-3
```

```
echo.
echo 2) Word 5.5
echo.
echo 3) Windows
echo.
.....
```

ECHO is designed for batch files, but you can use it from the command prompt as well. To suppress the DOS prompt temporarily, enter the command ECHO OFF. Bingo! The DOS prompt has gone. Enter ECHO ON and it comes back.

## EDIT

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

This command invokes a full-screen editor for viewing and editing ASCII (text) files. This command is ideal for use with BAT and INI files.

### Usage

`edit [[`*drive:*`][`*path*`]`*filename.ext*`] [/b] [/g] [/h] [/nohi]`

### Options and switches

When you use EDIT with no switches, the editor displays a welcoming dialog box.

`[[`*drive:*`][`*path*`]`*filename.ext*`]`

This option specifies the name of the ASCII file to be edited. If the file does not exist, a new, empty file is created.

`[/b]`

This switch suppresses colors and uses only black and white. Use this option on monochrome or LCD systems.

`[/g]`

This switch instructs DOS to use a special screen-writing technique to improve performance on CGA systems.

```
[/h]
```

On EGA systems, the editor switches into 43-line mode; on VGA systems and better, the editor switches to 50-line mode. The editor uses 25 lines by default.

```
[/nohi]
```

This switch instructs DOS to use the eight low-intensity colors.

## Remarks

Press F1 at any time to view EDIT's on-line help. If the help doesn't work, select **O**ptions Help **P**ath from the menu and enter the path to the file EDIT.HLP.

EDIT supports block operations such as cut, copy, and paste. To select a block of text, either hold down a Shift key and use the cursor keys, or press the left mouse button down and drag the mouse cursor to select the text. Search, replace, and print operations also are supported.

The editor can edit only one file at a time, and sufficient memory must exist for the entire file to be loaded into memory; no limit exists, however, on the length of each text line.

Please use the edit menu **O**ptions **D**isplay to change the text colors from boring gray on black to something a little more upbeat.

## Examples

To edit the CONFIG.SYS file, enter the following command:

```
edit c:\config.sys
```

To edit the WIN.INI file in 50-line mode on a VGA system, enter the following command:

```
edit c:\windows\win.ini /h
```

## Additional information

Chapter 10 includes some tips for using EDIT.

## Bob's Shareware Collection disks

As useful as EDIT is, it still cannot perform many editing tasks. If you want an easy-to-use but industrial-strength editor, try QEDIT, from SemWare. In addition to all the standard editing features, QEDIT supports editing of multiple files at one time, copying and pasting between files, column-oriented block operations, and a rich macro language, and it is fully customizable.

GEM

The DOS file EDIT.COM invokes QBASIC.EXE with a /EDITOR switch. If DOS cannot locate QBASIC (because it is not in the path or because you have deleted it), you cannot use EDIT.

BOB'S SCORECARD 3

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

# EGA.SYS

## Description

This device driver supports DOSSHELL task swapping on EGA systems.

## Usage

```
device[high]=[drive:][path]ega.sys
```

## Options and switches

`[`*drive:*`][`*path*`]`

This option identifies the directory in which EGA.SYS is located, if it is not in the boot directory. By default, SETUP places EGA.SYS in the \DOS directory.

## Remarks

If you use DOSSHELL for task swapping and your computer has an EGA video display, you must load the EGA.SYS device driver in CONFIG.SYS. EGA.SYS has no other uses.

EGA.SYS enables the DOSSHELL to maintain the video modes for every active session.

## Example

Use the following command in CONFIG.SYS to load the EGA.SYS device drivers:

```
devicehigh=c:\dos\ega.sys
```

## Additional information

CROSS REFERENCE

See Chapter 2 for a discussion of the DOSSHELL.

*See also* DEVICE, DEVICEHIGH, and DOSSHELL.

# EMM386

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ☐ | ☒ | ■ |

## Description

EMM386 is both a device drive and a command which provides advanced memory management capabilities on 386 systems and higher.

## Usage

You can execute EMM386 from the DOS prompt by using the following syntax:

```
emm386 [on or off or auto] [w=on or w=off]
```

When EMM386 is used as a device driver in CONFIG.SYS, it has the following bevy of switches:

```
device=[drive:][path]emm386.exe [on or off or auto] [w=on or
w=off] [memory] [min=size] [mx or frame=address or /pmmmm]
[pn=mmmm] [x=mmmm-nnnn] [i=mmmm-nnnn] [b=mmmm]
[l=minxms] [a=altregs] [h=handles] [d=nnn] [noems] [novcpi]
[highscan] [nohi] [verbose] [ram=mmmm-nnnn] [rom=mmmm-nnnn]
[win=mmmm-nnnn] [nomovexbda] [altboot]
```

So many switches, so little time.

## Options and switches

*Note:* Some switches, such as I= and X=, accept a memory range and have the syntax *mmmm–nnnn. mmmm* is the hexadecimal address of the first segment in the range, and *nnnn* is the hexadecimal address of the last segment in the range I=b000 - b7ff.

```
[drive:][path]emm386.exe
```

This option identifies the directory in which EMM386.EXE is located, if it is not in the boot directory. By default, SETUP places EMM386.EXE in the \DOS directory.

```
[on or off or auto]
```

This option controls whether EMM386 support for expanded memory and UMBs is active or dormant or in Auto Mode. By default, as you might expect, EMM386 is on. In Auto Mode, EMM386 provides EMS or UMB support only when such memory is requested by a program. You can turn off EMM386 only when the UMBs are not occupied and when no EMS memory is allocated. This switch is rarely used on most systems.

`[w=on or w=off]`

The Weitek brand math coprocessor can interfere occasionally with some DOS programs. Use the W= switch on systems equipped with the Weitek coprocessor. By default, support is off. This switch also is rarely used.

[*memory*]

This option specifies the maximum amount of extended memory EMM386 converts to expanded memory. The value is in kilobytes and must be in the range from 64K to 32768K. EMM386 uses a memory pool and converts extended memory to expanded memory only when a program requests expanded memory. When the program no longer needs the memory, it is returned to the pool. If the NOEMS switch is used, no expanded memory is supported. By default, EMM386 enables all XMS memory to be converted to EMS.

[min=*size*]

This switch sets the minimum amount of memory, in kilobytes, that EMM386 reserves as expanded. By default, EMM386 allocates 256K of memory as EMS memory when EMM386 is loaded without the NOEMS switch. To conserve extended memory, use MIN=0.

[mx or frame=*address* or /p*mmmm*]

You can use any one of these three switches to control where the EMS page frame is located in upper memory. The page frame occupies 64K of memory, which provides four 16K pages. Specify any value in the range from M1 to M14, to use the following memory locations:

| *Value* | *Location* | *Value* | *Location* |
|---|---|---|---|
| 1 | c000 | 8 | dc00 |
| 2 | c400 | 9 | e000 |
| 3 | c800 | 10 | 8000 |
| 4 | cc00 | 11 | 8400 |
| 5 | d000 | 12 | 8800 |
| 6 | d400 | 13 | 8c00 |
| 7 | d800 | 14 | 9000 |

Use switches m10 through m14 only on systems with 512K memory or more.

The FRAME= switch serves precisely the same function as M*X*, but you explicitly specify the memory address. Using M7, for example, is the same as using FRAME=d800. Only use addresses on the 400K boundary, as shown in the preceding table. The /P switch is the same as FRAME=.

`[pn=`*mmmm*`]`

Use this switch to set the specific address of each 16K page. LIM EMS 4.0 enables each page to be located in different memory locations (the 64K page frame does not have to occupy contiguous memory). *n* must be a value in the range from 1 to 4 (use p1=, p2=, p3=, and p4=). The p*n* switch supports values in the range from p0 to p255, where p0 to p3 are the standard four pages (you can allocate more than four pages). This switch is rarely used.

`[x=`*mmmm–nnnn*`]`

This switch instructs EMM386 not to use a specific area of upper memory (to *ex*clude it). *mmmm* must be lower than *nnnn,* and the values must be in the range from a000h to ffffh. This switch is used when EMM386 does not recognize that an area of upper memory is already occupied.

`[i=`*mmmm–nnnn*`]`

This switch instructs EMM386 to *i*nclude an area of upper memory it otherwise would avoid. *mmmm* must be lower than *nnnn,* and the values must be in the range from a000h to ffffh. This switch is commonly used on color video systems to include the unused monochrome video BIOS region (I=b000–b7ff, for example). If a memory area is explicitly excluded with X=, any contradictory attempts to include it with I= are ignored.

`[b=`*mmmm*`]`

This switch sets the lowest memory address used for EMS bank switching. Bank switching, standardized in EMS 4.0, enables programs such as DesqView to use expanded memory to manage multiple DOS sessions. The default value is 4000h. This switch is rarely used.

`[l=`*minxms*`]`

This switch, the converse of [*memory*], identifies the amount of extended memory to reserve (or not convert to expanded memory). This value is zero by default, which enables all extended memory to be converted to expanded memory.

`[a=`*`altregs`*`]`

Alternative register sets are used by programs, such as DesqView, that manage multiple DOS sessions. By default, EMM386 provides seven alternative register sets. If you run many DOS sessions concurrently, you can improve performance by increasing the number of registers, but you lose more than 200 bytes of base memory for every additional set. Values in the range from 0 to 254 are accepted.

`[h=`*`handles`*`]`

This switch sets the number of available EMS handles. The default is 62, but any value in the range from 2 to 255 is accepted.

`[d=`*`nnn`*`]`

This switch sets the amount of memory, in kilobytes, reserved for DMA buffers. By default, 16K is allocated, but values in the range from 16 to 256 are accepted.

`[noems]`

This popular switch is used to instruct EMM386 to provide UMB support but not expanded memory. When you use this switch, an additional 64K of upper memory is available as UMBs — this memory otherwise would be used for the EMS page frame.

`[novcpi]`

If NOEMS is used, you can use the NOVCPI switch to disable VCPI EMS support and save additional extended memory.

`[highscan]`

This option instructs EMM386 to use a rigorous technique to search all the upper memory for free space; this search can cause a lockup on some systems. Remove HIGHSCAN to stop EMM386 from searching all that upper memory. You can use the I= and X= switches to identify "manually" the memory EMM386 should use.

`[nohi]`

By default, EMM386 loads some of its code in upper memory. Specify NOHI to force EMM386 to load entirely in base memory.

[/verbose]

This switch instructs EMM386 to display detailed progress messages while EMM386 is loading.

[ram=*mmmm–nnnn*]

This option instructs EMM386 to only use the specified memory addresses for allocation as UMBs. *mmmm* must be lower than *nnnn,* and the values must be in the range from a000h to ffffh.

[rom=*mmmm–nnnn*]

This switch specifies the range of upper memory to be used as shadow RAM. *mmmm* must be lower than *nnnn,* and the values must be in the range from a000h to ffffh.

[win=*mmmm–nnnn*]

This switch instructs EMM386 to leave a range of upper memory so that Windows can use it. *mmmm* must be lower than *nnnns,* and the values must be in the range from a000h to ffffh.

[nomovexbda]

This switch instructs EMM386 not to move extended BIOS information from base to upper memory.

[altboot]

Use this switch if you experience problems rebooting the system using Ctrl-Alt-Del when EMM386 is loaded.

## Remarks

HIMEM.SYS must be loaded before EMM386.EXE is loaded.

In the majority of cases, you can use MEMMAKER, the DOS 6 memory optimizer, to fine-tune the EMM386 switches.

In earlier versions of DOS, the EMM386 default provided only EMS, and the RAM switch was used to explicitly instruct EMM386 to provide EMS *and* UMB support. With DOS 6, EMM386 defaults to provide EMS and UMB support.

Also, EMM386 now automatically includes the region c000:efff.

### Examples

The following extract from CONFIG.SYS loads EMM386 and optimizes upper memory by suppressing expanded memory support and utilizing the monochrome video BIOS area.

```
device=himem.sys
device=c:\dos\emm386.exe ram i=b000-b7ff noems
```

The following statement from CONFIG.SYS instructs EMM386 to emulate expanded memory and reserve an area of upper memory for Windows:

```
device=c:\dos\emm386.exe ram i=b000-b7ff win=b300-b7ff
```

CROSS REFERENCE

### Additional information

Memory management issues are discussed thoroughly in Chapter 5.

*See also* DOS, DEVICE, DEVICEHIGH, HIMEM.SYS, MEM, MEMMAKER, and SMARTDRV.

DISK

### Bob's Shareware Collection disks

EMM386 can be used on only 386 systems and higher. UMB_DRV.SYS, from Christopher Blum, additionally provides advanced memory management capabilities for 286 systems.

## ERASE

*See* DEL.

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ■ | ☐ |

## EXIT

### Description

This command terminates the current COMMAND.COM session and returns to the preceding COMMAND.COM session.

### Usage

```
exit
```

### Options and switches

None.

### Remarks

If you use Windows, you are accustomed to using EXIT to terminate a DOS session and return control to Program Manager. You use this command also when you *drop to DOS* from a DOS program such as 1-2-3 or Turbo Pascal. These programs launch a second copy of the command interpreter. After you execute EXIT, the secondary copy is terminated and you return to the application program.

### Example

To return from a Windows DOS box to Program Manager, enter the following command:

```
exit
```

### Additional information

*See also* COMMAND.

## EXPAND

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

DOS 6 is distributed in compressed files on the DOS disks. You must use EXPAND to transfer files manually from the DOS distribution disks to your system.

### Usage

```
expand [drive:][path]filename.ext [, [drive:][path][filename.ext [....]]
destination
```

## Options and switches

When you use EXPAND with no switches, it prompts for the source filename and the destination.

*[drive:][path]filename.ext*

This option specifies the name of the source file to be transferred from the DOS 6 master disks. You can specify multiple filenames when they are separated by spaces, but wildcards are not supported.

*destination*

This option specifies the drive or path to which the file is copied. If a single source file is specified, you can optionally specify a new filename for the target file. Normally, however, you should retain the original DOS filename.

## Remarks

Conceptually, EXPAND is similar to COPY except that EXPAND decompresses the source file before copying it to the destination.

When you first install DOS 6, SETUP transfers all the DOS files to your system. Use EXPAND if one or more of the DOS files is damaged or deleted.

## Examples

Use the following command to expand the file and copy it to the DOS subdirectory:

```
expand b:edit.hl_ c:\dos
```

The following command copies three files to the DOS subdirectory:

```
expand a:cga.vi_ a:cga.in_ a:cga.gr_ c:\dos
```

## Additional information

TIP

View the file PACKING.LST on the DOS Setup disk 1 to see which DOS 6 disk contains the files you want to restore. All compressed files have an underscore character at the end of the filename. If a file has no underscore, such as QBASIC.EXE, it is not compressed and can be transferred by using COPY or XCOPY.

## FASTHELP or DOSHELP

BOB'S SCORECARD 3

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

This command provides terse help on nearly all DOS commands but not on device drivers.

### Usage

```
fasthelp [command–name]
```

### Options and switches

When you execute FASTHELP with no switches, help is displayed for every DOS 6 command, and the display is paused after every screen.

```
[command–name]
```

This option specifies the name of the DOS command you want help with.

### Remarks

The help text that is displayed is stored in the ASCII file DOSHELP.HLP, located in the same directory as the FASTHELP program. You can modify this command to add your own help text for such items as batch files and utilities.

Help is available also by using the help command or by executing DOS utilities with the /? switch.

### Examples

To display help for the batch command MEM, enter the following:

```
fasthelp mem
```

To display a list of all DOS commands and a brief description, enter the following command:

```
fasthelp
```

### Additional information

CROSS REFERENCE

Refer to Chapter 2 to learn how to customize the help file.

*See also* HELP.

TIP

You can print a copy of the FASTHELP output by redirecting the output to the printer with the following command:

```
fasthelp > PRN
```

FASTHELP is smart enough not to pause for a keypress when the output is redirected.

## FASTOPEN

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 4 | ☒ | ☒ | ☒ | ☒ |

### Description

This command improves file performance by speeding access to frequently used files.

### Usage

Use the following syntax from a batch file or DOS prompt:

```
fastopen drive:[[=]n] ... [/x]
```

Alternatively, because FASTOPEN is a TSR, you can use INSTALL in CONFIG.SYS as follows:

```
install=[drive:][path] fastopen.exe drive:[=n] ... [/x]
```

### Options and switches

*drive:*

This option specifies the drive to be managed by FASTOPEN. You can specify multiple drives, but you must separate each one with a space.

`[[=]n]`

This option specifies the number of files on the drive on which FASTOPEN maintains statistics. Values in the range from 10 to 999 are accepted, and 48 is the default. Note that 48 bytes are allocated for every file.

`[/x]`

This switch instructs DOS to use expanded memory rather than base memory.

## Remarks

I do not recommend that you use this command. FASTOPEN is a poor person's disk cache. FASTOPEN just records details of where a file is stored when it is opened. If you open the file again later, DOS obtains the file details from memory rather than rereading the FAT. Tangible performance gains are realized only on systems that open and close files frequently (database applications, for example). FASTOPEN works only on local hard drives.

Far more performance gains are realized if you use SMARTDRV or a third-party disk cache — these programs cache the file data and the location details.

Do not load FASTOPEN from within the DOSSHELL or Windows or any other multitasker.

You must reboot your system to change the FASTOPEN configuration.

## Example

Use the following command to instruct FASTOPEN to monitor 50 files on drives C and D and store the details in expanded memory:

```
fastopen C:=50 d:=50 /x
```

## Additional information

CROSS REFERENCE

*See also* INSTALL and SMARTDRV.

CAUTION

Do not use DEFRAG or any other disk defragmenter if FASTOPEN is installed. You can lose valuable data.

# FC

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

## Description

FC, short for *f*ile *c*ompare, compares two files and highlights any differences.

## Usage

Use the following syntax when you compare two ASCII files:

```
fc [drive:][path]filename1.ext [drive:][path]filename2.ext [/a] [/c]
[/l] [/lbn] [/n] [/t] [/w] [/nnnn]
```

Use the following syntax when you make a binary comparison of files:

```
fc [drive:][path]filename1.ext [drive:][path]filename2.ext [/b]
```

## Options and switches

*[drive:][path]filename1.ext*

This option specifies the name of the first file to be compared.

*[drive:][path]filename2.ext*

This option specifies the name of the second file to be compared. FC accepts a wildcard as the second filename, but normally you specify the individual filename.

`[/a]`

This switch specifies that an *a*bbreviated list of differences should be displayed when you are comparing ASCII files. Only the first and last lines in each set of differences will be displayed.

`[/c]`

This switch instructs FC to ignore the case of letters.

`[/l]`

The /L switch instructs FC to use a line-by-line comparison to identify the location of blocks of lines missing from a file, and to try to resynchronize and find the next common line. This switch is used by default when the filename does not have an extension of BIN, COM, EXE, LIB, OBJ, or SYS.

`[/lbn]`

This switch sets the internal line buffer. By default, 100 lines are reserved to store a missing block from one of the files. If FC fails and returns this message:

```
Resync failed. Files are too different
```

try using the /LB*N* switch to increase the buffer size.

`[/n]`

This switch instructs FC to display line numbers in comparing ASCII files.

```
[/t]
```

By default, FC treats a tab character as though it were eight spaces during an ASCII compare. This switch instructs FC to use the Tab character for the comparison and not to expand the tabs with spaces.

```
[/w]
```

This switch instructs FC to treat contiguous white-space characters (spaces and tabs) as a single space. Use this switch when you are comparing two ASCII files and you don't care about differences in the spaces and tabs.

`[/`*nnnn*`]`

FC tries to resynchronize ASCII files when one or more lines is encountered in one file and not in the other. This switch specifies how many lines must be identical before FC assumes that the files are resynchronized. By default, the value is 2.

```
[/b]
```

This switch instructs FC to perform a binary (byte-by-byte) comparison of the files. In this mode, FC does not attempt to resynchronize after a problem occurs. This mode is assumed when you are comparing files with the following extensions: BIN, COM, EXE, LIB, OBJ, or SYS.

## Remarks

Use | MORE to pause after each screen.

If you use the /B switch to perform a binary comparison, do not use any other switches (specify only the filenames).

When a difference is found during a binary compare, FC displays, in hexadecimal, the position from the beginning file, the byte value in file 1, and the byte value in file 2.

## Examples

Consider the following two ASCII files that detail omelette recipes:

RECIPE.ONE

```
two eggs
dash of milk
half diced tomato
tablespoon cheese
half diced green pepper
```

RECIPE.TWO

```
two eggs
dash of milk
pinch salt
two cups jalapeno
half diced tomato
tablespoon cheese
half diced chile pepper
```

You can use the following command to compare the two files in ASCII mode and display line numbers:

```
fc recipe.one recipe.two /n
```

The following output is generated:

```
Comparing files RECIPE.ONE and RECIPE.TWO
***** RECIPE.ONE
    2:  dash of milk
    3:  half diced tomato
***** RECIPE.TWO
    2:  dash of milk
    3:  pinch salt
    4:  two cups jalapeno
    5:  half diced tomato
*****

***** RECIPE.ONE
    4:  tablespoon cheese
    5:  half diced green pepper
***** RECIPE.TWO
    6:  tablespoon cheese
    7:  half diced chile pepper
*****
```

## Additional information

*See also* DISKCOMP.

BOB'S SCORECARD 3

PROMPT ☐ AUTOEXEC ☐ BATCH ☐ CONFIG ■

# FCBS

## Description

This command specifies the maximum number of file control blocks DOS can have open at any time.

### Usage

FCBS is used in CONFIG.SYS with the following syntax:

```
fcbs=x
```

### Options and switches

*x*

This option identifies the maximum number of file control blocks. Any value in the range from 1 to 255 is accepted. If CONFIG.SYS has no FCBS entry, DOS assigns 4.

### Remarks

FCBS, a hand-me-down from DOS 1, was superseded by the FILES= statement in DOS 2. Few (useful) programs use the outdated FCBS file-management technique.

### Example

To save memory, add the following FCBS statement to CONFIG.SYS:

```
fcbs=1
```

CROSS REFERENCE

### Additional information

Refer to Chapter 5 for other memory-saving techniques.

*See also* FILES.

## FDISK

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

This command prepares a new hard disk for formatting and manages logical drive assignments on a partitioned disk.

### Usage

```
fdisk [/status]
```

### Options and switches

Used with no switches, FDISK presents a full-screen menu for configuring the physical hard disks.

```
[/status]
```

This switch instructs FDISK to display a brief summary of the system's hard drive configuration. The configuration summary cannot be piped to a file or device.

## Remarks

FDISK is responsible for storing the disk partition information on the hard disk. This information is stored in the *master partition table,* located in the first sector of the first track on the first platter. When you first boot the computer, the partition table is inspected and attention turns to the boot partition from which the appropriate operating system is loaded. The partition used to boot the computer is referred to as the *active partition.*

DOS recognizes two types of partitions: the *primary* partition and the *extended* partition. Every DOS system must have a primary partition. If you want the entire disk to be a single logical drive (typically drive C), you need only a primary partition. If you want to divide the disk into multiple DOS partitions, however, you must create an extended partition also. The extended partition is used to record as many as 23 additional logical drives. The primary partition should be identified as the active partition if you want your computer to boot up under DOS. The primary partition contains the DOS boot files IO.SYS, MSDOS.SYS, and COMMAND.COM.

To change the size of a logical drive (a partition), you must delete the old partition and create a new one.

## Examples

Use the following command to run the FDISK program, which presents you with the main menu:

```
fdisk
```

To see your current partition status, enter the following command:

```
fdisk /status
```

## Additional information

MSD provides additional information about your hard disk configuration. Refer to Chapter 4 for a thorough discussion of hard disk management.

*See also* FORMAT, MSBACKUP, and MSD.

CAUTION

*FDISK can damage your health.* If you are using FDISK on a system that is already partitioned and that has useful data, be careful. Destroying a partition and losing all the data in the partition is too easy. Make sure that you have a complete, up-to-date backup of your entire system before you make any changes to the partition structure. Unlike most DOS commands, you should not experiment with this one.

# FILES

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

## Description

This command specifies the number of files that DOS can have open concurrently.

## Usage

```
files=nn
```

## Options and switches

*nn*

This option specifies the maximum number of open files. Any value in the range from 8 to 255 is accepted. When CONFIG.SYS does not include a FILES command, DOS uses 8.

## Remarks

You might think that you never will need more than eight files open at a time. That situation may have existed five years ago, but things certainly have changed. If you use Windows, consider setting FILES to 40 or more; otherwise, a value of 20 to 30 is appropriate.

Each additional file handle you specify with FILES consumes 60 bytes.

Most well-behaved programs display a meaningful message if a file cannot be opened because of a shortage of file handles. If you encounter this type of message, increase the FILES setting by 5, and try again. You should not need a FILES allocation larger than 50.

## Example

Use the following command in CONFIG.SYS to enable DOS to have 40 open files:

```
files=40
```

## Additional information

*See also* FCBS.

BOB'S SCORECARD 6

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **FIND** | ■ | ■ | ☒ | ☐ |

## Description

This command searches for text in one or more files.

## Usage

`find "string" [[drive:][path]filename.ext ...] [/v] [/c] [/n] [/i]`

## Options and switches

*"string"*

This option specifies the string to search for, enclosed in double quotation marks. If the string you want to search for includes a quote character, specify two quotation-mark characters together so that DOS interprets it as one quotation mark within the string.

`[[drive:][path]filename.ext ...]`

This option specifies the file in which to search for the text. Wildcards are not supported, but you can specify multiple filenames.

`[/v]`

This switch instructs FIND to identify all the lines that do *not* include the string.

`[/c]`

This switch instructs FIND to display the total number of lines where the string was found (the count).

`[/n]`

This switch displays a line number with each line.

`[/i]`

This switch instructs FIND to ignore the character case (a case-insensitive search).

## Remarks

FIND is designed for ASCII files, and the search string is noticed only if it is situated on a line (the string cannot extend beyond a line break).

The DOS editor, EDIT, includes a search and a search-and-replace facility.

If you do not specify a filename, FIND interprets all the characters you type on the command line searching for the string. This effectively disables normal command input. Just press Ctrl-Break to return to the DOS prompt.

## Examples

Use the following command to search for the text *goto* in AUTOEXEC.BAT and display line numbers on matching lines:

```
find “goto” autoexec.bat /n
```

I generated the following output on my system:

```
---------- AUTOEXEC.BAT
[6]goto %CONFIG%
[10]goto QUIT
[15]if errorlevel 3 goto continue
[16]if errorlevel 2 goto single
[18]goto continue
```

## Additional information

See the FOR command for an example of using FIND on multiple files.

*See also* FOR and EDIT.

## Bob’s Shareware Collection disks

SeakEasy, from Correlation Systems, is an excellent full-screen text-finding utility. It can search multiple files with wildcards, and even search the current directory, all nested subdirectories, or the entire drive.

The versatile PocketD program from Pocketware includes text searching options that you can invoke from the command line by using switches.

You can pipe the output from the DIR command to FIND, which is an easy way to search for filenames that contain a specific character string.

To find all files on drive C that include the characters *read*, for example, enter the following command:

```
dir c:\*.* /s /b | find “read” /i
```

The /B switch improves performance because FIND must search only the filenames and not the file size and date of creation, for example. When the /B switch is omitted, you can use FIND to show all the files created or modified on a certain date. The following command searches for all the files modified on March 31, 1993:

```
dir c:\*.* /s | find “03-31-93”
```

***Note:*** If you ask FIND to look for several words separated by spaces, it will fail to recognize them if the words get split up across two lines.

# FOR

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ☒ | ■ | ☐ |

## Description

This command executes a command or statement multiple times and applies a new string or filename with each iteration.

## Usage

When FOR is used in a batch file, the syntax is as follows:

```
for %%variable in (set) do command [command–parameters]
```

When you execute FOR from the DOS prompt, the syntax is as follows:

```
for %variable in (set) do command [command–parameters]
```

***Note:*** The difference is the double % symbol in batch files.

## Options and switches

*%%variable* and *%variable*

A single character variable identifier that represents the active value in the *set*. The variable value is updated with the next value in the *set* with each iteration of the FOR loop. The variable is represented by a single character, but the following characters are not allowed:

0 through 9 / | < >

The same variable identifier often is used in the command or the command parameters.

(*set*)

This option specifies a list of one or more strings or filenames separated by spaces. With each iteration of the loop, a value from the list is used in place of the *variable* identifier.

*command*

This option specifies the command to be executed with each iteration of the loop.

[*command-parameters*]

This option specifies any parameters or switches the command requires.

## Remarks

You can use wildcards when filenames are used in the (*set*).

## Examples

The following batch file command lists all the programs in the current directory:

```
@echo off
for %%n in (*.EXE *.COM *.BAT) do echo %%n
```

In the preceding example, the variable name acts as a placeholder, and the name that is used is not important. The same results can be achieved, for example, with the following command (the only difference is the variable identifier name):

```
@echo off
for %%a in (*.EXE *.COM *.BAT) do echo %%a
```

When you combine FOR with FIND, you can use FOR to search for files that contain some specific text. When the following command is executed from the DOS prompt, it searches all TXT and ME files for the string:

```
for %s in (*.txt *.me) do find “help” %s.
```

**NOTE** Only EXEs and COMs work as commands. If you want to repeatedly execute a batch file, be sure to use the CALL command to call the batch file. For example, the following statement will execute all the batch files in the root directory:

```
@echo off
for %b in (\*.bat) do call %b
```

**CROSS REFERENCE**

## Additional information

Refer to Chapter 10 for some more examples of using FOR in batch files.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **FORMAT** | 7 | ■ | ☐ | ☒ | ☐ |

### Description

This command formats a floppy or hard disk and creates a new file allocation table; optionally marks bad sectors.

### Usage

***Note:*** A high-density disk drive is capable of formatting disks with different capacities. The FORMAT command provides several different switches you can use to achieve the same FORMAT results. Consequently, you can use FORMAT in any of the following ways:

```
format drive: [/v[:label]] [/q] [/u] [/s] [/f:size]

format drive: [/v[:label]] [/q] [/u] [/s] [/t:tracks /n:sectors]

format drive: [/v[:label]] [/q] [/u] [/s] [/1] [/4]

format drive: [/q] [/u] [/s] [/1] [/4] [/8]
```

### Options and switches

*drive:*

This option specifies the drive to be formatted.

`[/v[:label]]`

This switch instructs FORMAT to assign a volume label to the newly formatted disk. If you use /V without the new label, FORMAT prompts you to enter the new label after the drive has been formatted. The label can be 11 characters long and is converted automatically to uppercase.

`[/q]`

This switch instructs FORMAT to perform a quick format. This mode provides a fast way to reformat disks that are in good condition. In Quick Mode, the FAT and root directories are cleared, but the disk sectors are not reformatted — think of it as a fast way to erase all the files and subdirectories on a disk.

`[/u]`

FORMAT normally records key file information behind the scenes before formatting a disk. UNFORMAT accesses this information when you try to undo an accidental format. The /U command forces an unconditional format, which means that the disk cannot be unformatted. Use the switch only when you are experiencing read or write errors with the disk.

`[/s]`

This switch makes a disk bootable by copying the operating system files IO.SYS, MSDOS.SYS, COMMAND.COM, and DBLSPACE.BIN.

The remaining switches control the capacity of formatted disks.

`[/f:size]`

I think that this switch provides the easiest way to specify the size of the disk. Use one of the following numbers:

| ***Size Number*** | ***Disk size*** |
|---|---|
| 160 | A 160K single-sided double-density 5¼-inch disk |
| 180 | A 180K single-sided double-density 5¼-inch disk |
| 320 | A 320K double-sided double-density 5¼-inch disk |
| 360 | A 360K double-sided double-density 5¼-inch disk |
| 720 | A 720K double-sided double-density 3½-inch disk |
| 1200 | A 1.2MB double-sided high-density 5¼-inch disk |
| 1440 | A 1.44MB double-sided high-density 3½-inch disk |
| 2880 | A 2.88MB double-sided super-high-density 3½-inch disk |

The *size* number optionally can be followed with K or KB (720, 720K, and 720KB, for example, all have the same meaning).

`[/t:`*tracks* `/n:`*sectors*`]`

This switch explicitly identifies the number of tracks and the number of sectors to apply during the format.

`[/1]`

This old switch is rarely used because it creates single-sided disks.

`[/4]`

This switch formats a 360K 5¼-inch disk in a 1.2MB drive.

`[/8]`

This old switch is rarely used either because it creates the now obsolete 320K floppies used in DOS 1.1.

## Remarks

You can use the UNFORMAT command to undo an accidental disk format if the /U switch was not specified. See the caution later in this section.

The now obsolete /B switch is still supported. /B was required with DOS versions before DOS 4 and was used to reserve space for system files. Thanks to /S, the /B switch no longer is required, but it is preserved for backward compatibility.

A disk volume label can be changed subsequently with the LABEL command.

When FORMAT is used in batch files, it returns one of the following ERRORLEVELs:

| *ERRORLEVEL* | *Message* |
|---|---|
| 0 | The disk was formatted successfully. |
| 3 | The user aborted the format by pressing Ctrl-Break. |
| 4 | An unexpected error occurred! |
| 5 | The user entered N when prompted to confirm the FORMAT operation. |

## Examples

Use the following command to format a disk in drive A, assuming that the disk and the drive capacities are the same:

```
format a:
```

Use the following command to format a 360K disk in a 1.2MB drive:

```
format b: /s:360
```

Use the following command to create a bootable disk and add the volume label TECHNOJOCK:

```
format a: /v:technojock /s
```

CROSS REFERENCE

## Additional information

*See also* DELTREE, FDISK, LABEL, and UNFORMAT.

DISK

## Bob's Shareware Collection disks

The New-Ware program Format Master provides full-screen formatting tools for formatting floppy disks. If you are formatting disks to make copies of other disks, try DUPE, from Best Technology. It can format and copy a disk much faster than DOS, and it's easy to use.

CAUTION

Use the FORMAT command with caution. Although you can use it to undo a format, it is not foolproof, and valuable data may get lost. UNFORMAT cannot recover fragmented files, for example. Always inspect a drive before formatting it to verify that you no longer need the files.

UNDOCUMENTED FEATURE

The following three undocumented switches are supported by FORMAT:

```
/backup
```

This switch bypasses the `Insert new diskette` prompt.

```
/select
```

This switch does not format the disk at all, but creates the UNFORMAT files MIRROR.FIL and MIRRORSAV.FIL.

`/autotest`

Use this switch to quickly format a single diskette. The `Insert new diskette` prompt, the prompt for a volume label, and the prompt `Format another (Y/N)` are all bypassed.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **GOTO** | 7 | ☐ | ■ | ■ | ☐ |

## Description

This command instructs DOS to continue processing batch file commands at a specific location in the batch file.

## Usage

`goto` *label*

## Options and switches

*label*

This option identifies the next location in the batch file where commands should be processed. The label is not case sensitive, and as many as eight characters are supported. The label can include spaces, but not the characters <, >, [, ], %, *, –, +, =, \, /, . , ,; ,:, or |.

## Remarks

A colon must precede the target location, *label*. If a batch file includes the statement `goto quit`, for example, another line in the batch file must have the contents `:quit`.

You can specify batch file parameters %0 through %9 as the GOTO label.

## Example

The following batch file uses GOTO in conjunction with IF and ERRORLEVEL to load a device driver:

```
@echo off
CHOICE /c:mtn Do you want to load the Mouse, Tablet or Neither
if errorlevel 3 goto end
if errorlevel 2 goto tablet
mouse /y
goto end
:tablet
tabk /c:345
:end
```

### Additional information

Chapter 10 includes numerous examples of batch files using GOTO.

*See also* CHOICE, ERRORLEVEL, and IF.

### Bob's Shareware Collection disks

TurboBat, with the ExtraDOS Toolbox from Foley Hi-Tech Systems, compiles batch files to improve performance and prevent other people from tampering with the code.

## GRAPHICS

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ☒ | ☐ |

### Description

This command enables you to print graphics screens just by pressing PrtScr on CGA, EGA, and VGA systems.

### Usage

graphics [*type*] [[*drive:*][*path*]*filename.ext*] [/r] [/b] [/lcd] [/printbox:std or /printbox:lcd]

### Options and switches

[*type*]

The instructions used to print graphics vary from printer to printer. You use this switch to identify the printer type.

The following printer types are supported:

| | |
|---|---|
| COLOR1 | IBM PC Color printer with single color ribbon |
| COLOR4 | IBM PC Color printer with a RGB ribbon |
| COLOR8 | IBM PC Color printer with a CMY ribbon |
| DESKJET | HP Deskjet printer |
| GRAPHICS | IBM Personal Graphics, Proprinter, or Quietwriter printer |
| GRAPHICSWIDE | 11-inch model of IBM Personal Graphics |
| HPDEFAULT | All HP PCL printers |
| LASERJET | Original HP LaserJet printer |
| LASERJETII | HP LaserJet Series II printer |
| QUIETJET | HP QuietJet printer |
| QUIETJETPLUS | HP QuietJet Plus printer |
| RUGGEDWRITER | HP RuggedWriter printer |
| RUGGEDWRITERWIDE | HP RuggedWriterwide printer |
| THERMAL | IBM PC-Convertible Thermal printer |
| THINKJET | HP ThinkJet Printer |

[[*drive:*][*path*]*filename.ext*]

This option specifies the name of a graphics profile file that contains the printer escape codes (see the SECRET later in this section). DOS uses the file GRAPHICS.PRO by default in the DOS directory.

`[/r]`

This switch instructs DOS to print the image in reverse (black on-screen is black on the printer). By default, DOS prints white (or any other color) as black.

```
[/b]
```

This switch is used on color printers to print the background in color.

```
[/lcd]
```

The /LCD switch instructs DOS to use the same aspect ratio as an LCD display rather than a CGA ratio. This switch is the same as /PRINTBOX:LCD.

```
[/printbox:std or /printbox:lcd]
```

This switch controls the aspect ratio of the printed image. STD is used by default. Use /LCD — it's shorter.

### Remarks

You can optionally load the GRAPHICS TSR, which uses about 6K of memory, into upper memory with LOADHIGH.

PostScript printers are not supported.

### Example

To use the default settings to install graphics print-screen support, enter the following command:

```
graphics
```

### Additional information

CROSS REFERENCE

*See also* LOADHIGH and PRINT.

GEM

The ASCII file GRAPHICS.PRO contains all the printer control codes and escape sequences for the supported printers. If your printer is not supported, copy the file to CUSTOM.PRO and use the codes in your printer manual to adjust the codes for one of the existing printers. Then install GRAPHICS and specify CUSTOM.PRO as a switch.

## HELP

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

DOS 6

### Description

This command displays full-screen on-line help for DOS commands and device drivers.

## Usage

```
help [topic] [/b] [/g] [/h] [/nohi]
```

## Options and switches

When you use HELP with no switches, it displays a full-screen index of all the DOS commands and device drivers.

`[topic]`

This option specifies the name of the command or device driver you want help with. If the switch is not specified, or if the topic is unknown, the main help index is displayed.

`[/b]`

This switch suppresses colors and uses only black and white. Use this option on monochrome or LCD systems.

`[/g]`

This switch instructs DOS to use a special screen-writing technique to improve performance on CGA systems.

`[/h]`

On EGA systems, HELP switches to 43-line mode; on VGA systems and better, it switches to 50-line mode. HELP uses 25 lines by default.

`[/nohi]`

This switch instructs DOS to use the eight low-intensity colors.

## Remarks

I use this great command all the time.

To quit from on-line help and return to the DOS prompt, press Alt-F,X.

In addition to specifying the standard commands and device drivers, you can specify the following additional subtopics or categories:

| *Command* | *Help Topic* |
|---|---|
| batch | Batch file commands |
| config | CONFIG.SYS commands |
| international | Commands for international configurations |
| multi-config | Commands for creating a multiboot CONFIG.SYS menu |

Many commands also display help when you use the /? switch.

## Examples

To display help on HIMEM.SYS, enter the following command:

```
help himem.sys
```

To get help about batch file commands, enter the following command:

```
help batch
```

To display the main help index (or contents), enter the following command:

```
help
```

CROSS REFERENCE

## Additional information

Refer to Chapter 2 for additional information about the DOS help facilities.

*See also* FASTHELP.

# HIMEM.SYS

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

## Description

This device driver is responsible for managing extended memory on 286 systems and higher.

## Usage

```
device=[drive:][path]himem.sys [/a20control:on or off]
[/cpuclock:on or off] [/eisa] [/hmamin=m] [/int15=xxxx]
[/numhandles=n] [/machine:xxxxx] [/shadowram:on or off]
[/verbose]
```

## Options and switches

You usually use HIMEM.SYS with no switches. The majority of the switches are for use on nonstandard hardware.

[*drive:*][*path*]

This option identifies the directory in which HIMEM.SYS is located, if it is not in the boot directory. By default, SETUP places HIMEM.SYS in the \DOS directory.

`[/a20control:on` or `off]`

The A20 pin on 286 CPUs and higher is used to provide access to the high memory area (HMA); it must be managed by only one program or device driver. By default, HIMEM.SYS takes control of the a20 handler if it is set to OFF when HIMEM.SYS is loaded (it applies the switch /A20CONTROL:ON). To load HIMEM.SYS and instruct it not to handle the a20 pin, use the switch /A20CONTROL:OFF.

`[/cpuclock:on` or `off]`

On some systems, HIMEM.SYS can slow overall system performance significantly. If this slowdown occurs, use the switch /CPUCLOCK:ON to return the system to its original speed (HIMEM.SYS will function more slowly). By default, HIMEM.SYS uses /CPUCLOCK:OFF.

`[/eisa]`

Use this switch only on EISA bus systems with more than 16MB of RAM to instruct HIMEM.SYS to manage all extended memory; otherwise, HIMEM.SYS may manage only the first 16MB.

`[/hmamin=`*m*`]`

Only one application, usually the DOS kernel, can use the HMA. You can use the /HMAMIN switch to stop a small application from using the HMA. Use this switch to instruct HIMEM.SYS to give HMA control only to applications that request a minimum of *m* kilobytes of memory.

`[/int15=`*xxxx*`]`

Some older extended memory applications do not adhere to the XMS specification and use interrupt 15h to allocate extended memory. Use the /INT15 switch to reserve *xxxx* kilobytes of extended memory for servicing interrupt 15h requests. *xxxx* must be a value in the range from 64 to 65535, and it must not be greater than the total amount of installed memory. By default, HIMEM.SYS does not reserve any memory for these outdated programs.

`[/numhandles=n]`

Extended memory is managed in blocks (EMBs), and HIMEM.SYS by default can manage 32 EMBs. Use the /NUMHANDLES switch to specify a different value in the range from 1 to 128. /NUMHANDLES has no effect when running Windows in 386 enhanced mode.

`[/machine:xxxxx]`

HIMEM.SYS performs hardware-specific tasks that vary from computer to computer. When HIMEM.SYS is loaded, it checks to determine the computer type. If HIMEM.SYS does not recognize the computer model, it assumes that it is an IBM PC-AT. You can use the /MACHINE switch to explicitly identify the host computer brand. *xxxxx* can be one of the strings or numbers in the following table:

| **Number** | **String** | **Computer model** |
|---|---|---|
| 1 | at | IBM PC-AT or 100 percent compatible |
| 2 | ps2 | IBM PS/2 |
| 3 | ptlcascade | Phoenix Cascade BIOS |
| 4 | hpvectra | HP Vectra (A and A+) |
| 5 | att6300plus | AT&T 6300 Plus |
| 6 | acer1100 | Acer 1100 |
| 7 | toshiba | Toshiba 1600 and 1200XE |
| 8 | wyse | Wyse 12.5 MHz 286 |
| 9 | tulip | Tulip SX |
| 10 | zenith | Zenith ZBIOS |
| 11 | at1 | IBM PC AT (alternative delay) |
| 12 | at2 | IBM PC AT (alternative delay) |
| 12 | css | CSS Labs |
| 13 | at3 | IBM PC AT (alternative delay) |
| 13 | phillips | Phillips |
| 14 | fasthp | HP Vectra |
| 15 | ibm7552 | IBM 7552 Industrial Computer |
| 16 | bullmicral | Bull Micral 60 |
| 17 | dell | Dell XBIOS |

`[/shadowram:on` or `off]`

This switch can be used on some systems equipped with shadow RAM (systems that copy ROM code into faster RAM. This switch can save some extended memory (for applications) and is designed for systems with less than 2MB of extended memory. Use the switch /SHADOWRAM:ON to leave shadow RAM active, and /SHADOWRAM:OFF to disable it. By default, HIMEM.SYS attempts to disable shadow RAM on systems with 2MB or less of extended memory; otherwise, it leaves it on.

```
[/verbose]
```

This switch instructs HIMEM.SYS to display additional data as it loads.

## Remarks

HIMEM.SYS (or a third-party extended memory manager) must be installed before DOS can be loaded high and before EMM386 can be installed.

MEMMAKER automatically adds a HIMEM.SYS statement to CONFIG.SYS.

## Example

On most systems, you install HIMEM.SYS with no switches on the first line of CONFIG.SYS as follows:

```
device=c:\dos\himem.sys
```

## Additional information

CROSS REFERENCE

Refer to Chapter 5 for a thorough discussion of DOS 6 memory management issues.

*See also* DEVICE, DOS=, EMM386, MEM, and MEMMAKER.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **IF** | 7 | ☐ | ☒ | ■ | ☐ |

## Description

This command provides a conditional branch in a batch file execution.

## Usage

You can use IF to compare two strings, test the ERRORLEVEL a program returns, or test whether a file exists by using one of the following three syntax formats:

`if [not]` *string1==string2 command*

`if [not] errorlevel` *number command*

`if [not] exist [`*drive:*`][`*path*`]`*wildcard.ext command*

## Options and switches

`[not]`

The IF statement normally executes *command* when the test condition is true. Use the NOT operand when you want *command* to be executed when the test condition is false.

*command*

This option specifies any valid DOS or batch command.

*string1==string2*

This option returns a true value when the two strings are identical. To test for an empty string, use quotation marks to surround both values (see the "Examples" section below). The test is case sensitive.

`errorlevel` *number*

This option returns a true value when the ERRORLEVEL returned by the last command is equal to or greater than the specified *number*.

`exist [`*drive:*`][`*path*`]`*wildcard.ext*

This option returns a true value when one or more matching files exist.

## Remarks

You can use batch file variables (%1 through %9) in place of strings, ERRORLEVEL numbers, or filenames.

Always remember that the ERRORLEVEL test returns true if the *number* is *less than* or equal to the ERRORLEVEL. For example, ERRORLEVEL 0 always returns true because the 0 always is less than or equal to the ERRORLEVEL. You must test for multiple ERRORLEVELs in reverse order (see the second example in the following "Example" section).

## Examples

The following batch file adds extensive error checking to improve on the RENAME command and illustrates many uses of the IF statement:

```
@echo off
rem A batch file to test for file before renaming it
if “%1”==”” goto novar
if “%2”==”” goto novar
if “%1”==”%2" goto duplicate
if not exist %1 goto nosource
if exist %2 goto oldtarget
rename %1 %2 > nul
if not exist %2 goto failed
echo File Renamed!
goto quit
:duplicate
echo Source and Target filenames must be different
goto quit
:failed
echo Target file not created
goto quit
:nosource
echo Cannot Find the source file
goto quit
:oldtarget
echo The Target file already exists
goto quit
:novar
echo You must specify a source and target filename
:quit
```

The following batch file illustrates that you must test ERRORLEVEL in reverse order:

```
@echo off
CHOICE /c:mtn Do you want to load the Mouse, Tablet or Neither
if errorlevel 3 goto end
if errorlevel 2 goto Tablet
mouse /y
goto end
:Tablet
tabk /c:345
:end
```

### Additional information

Chapter 10 shows many ways you can use an IF statement in batch files.

*See also* CHOICE and GOTO.

### Bob's Shareware Collection disks

TurboBat, with the ExtraDOS Toolbox from Foley Hi-Tech Systems, compiles batch files to improve performance and prevent other people from tampering with the code.

## INCLUDE

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This command is used in a multiboot CONFIG.SYS file to instruct DOS to process a specific block of commands.

### Usage

```
include=blockname
```

### Options and switches

*blockname*

This option specifies the name of the configuration block whose commands are executed.

### Remarks

Starting with Version 6, DOS supports multiple configurations in CONFIG.SYS. A set of menu commands is used to define a boot-up menu, and DOS branches to a specific block based on a user's selection.

The INCLUDE statement provides a convenient way for multiple configurations to share a common set of commands without duplicating them.

After processing the commands, DOS returns to the command immediately following the INCLUDE statement.

You can nest INCLUDE blocks as much as ten levels deep (an INCLUDE block can include another block, which in turn can include another block, and so on).

When a block is named `[common]`, it is included automatically in all blocks.

### Example

The following CONFIG.SYS file includes the block `[basics]` in both configuration menu options.

```
[menu]
menuitem=justmouse,Just the Mouse
menuitem=gainers,Big Gainers
menudefault=gainers,5

[basics]
files=30
buffers=15

[justmouse]
include basics
device=c:\windows\mouse.sys /y

[gainers]
include basics
device=c:\dos\himem.sys
device=c:\dos\emm386.exe ram i=b000-b7ff noems
device=c:\dos\smartdrv.exe
dos=high,umb
devicehigh=c:\windows\mouse.sys /y
```

### Additional information

CROSS REFERENCE

Multiple configurations are discussed in depth in Chapter 8.

*See also* MENUCOLOR, MENUDEFAULT, MENUITEM, and SUBMENU.

## INSTALL

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This command loads memory-resident programs from CONFIG.SYS and avoids the allocation of unnecessary environment space.

### Usage

`install=[`*drive:*`][`*path*`]`*programname* `[`*program parameters*`]`

### Options and switches

[*drive:*] [*path*] *programname*

This option specifies the name of the memory-resident program.

[*program parameters*]

This option specifies the program's optional parameters and switches.

### Remarks

Not all memory-resident programs function correctly when you load them with INSTALL.

Each INSTALL statement is processed after all the device drivers have been installed, regardless of the location of the INSTALL statement in CONFIG.SYS.

On 386 systems and higher, you should load programs into UMBs with LOADHIGH rather than into base memory with INSTALL.

### Example

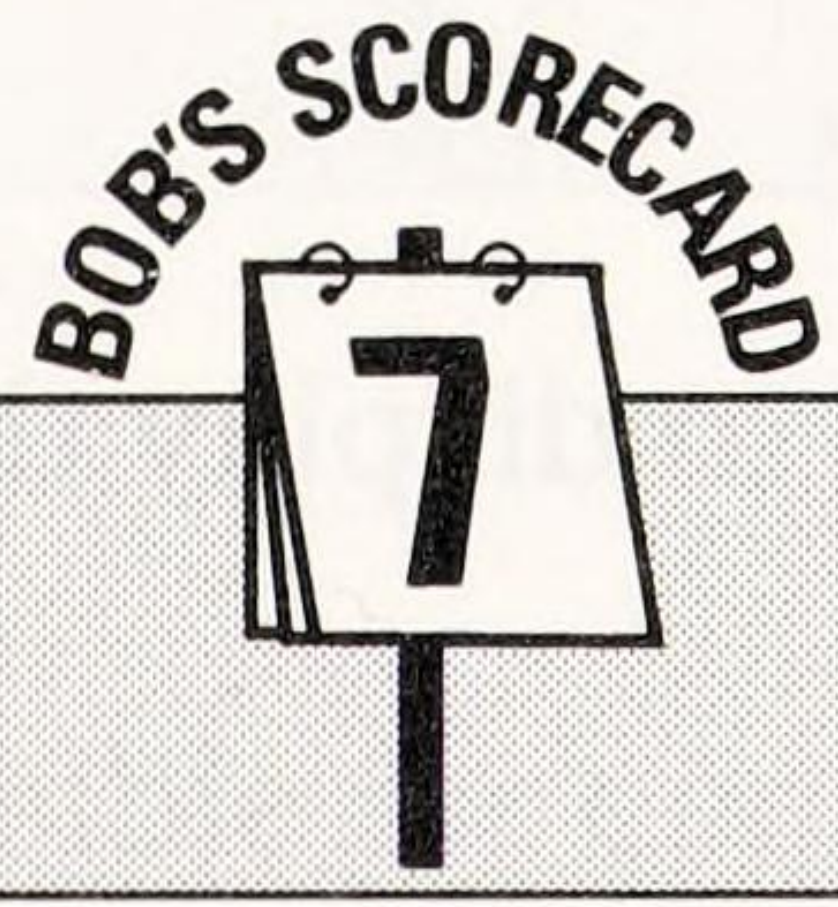
Use the following statement in CONFIG.SYS to install KEYB (the DOS keyboard manager):

```
install=c:\dos\keyb gr /e
```

### Additional information

*See also* LOADHIGH.

## INTERLNK



| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

This command changes the drive assignments for two computers connected by way of InterLink.

INTERLNK.EXE is a dual-purpose program: You must first use it as a device driver in CONFIG.SYS, and then you can execute it as a program from the DOS prompt. See the following INTERLNK.EXE section for a description of the syntax when INTERLNK.EXE is used as a device driver.

## Usage

```
interlnk [client[:]=[server][:]]
```

## Options and switches

When you use INTERLNK without switches, it displays the status of the active InterLink drive assignments.

*client*[:]=

This option specifies the letter of the drive on the local (or client) system that is to represent a drive on the server. The drive must have been used in the initial configuration when the two computers were first linked. The colon (:) is optional but should be used for clarity.

[*server*][:]

This option specifies the name of the host drive on the InterLink server that is accessed when the client drive letter is used. If you use this switch, you terminate the client drive connection. The colon (:) is optional but should be used for clarity.

## Remarks

To connect two computers, the client must load INTERLNK.EXE as a device in CONFIG.SYS, and the server must be running the INTERSVR command.

A LASTDRIVE statement must also be added to the CONFIG.SYS file if you want to assign a drive letter greater than D to one of the remote computer's drives.

## Examples

Use the following command to display the status of all connected drives:

```
interlnk
```

To assign the drive letter D to the server's C drive, enter the following command:

```
interlnk d:=c:
```

To disconnect the drive D assignment on the local (client) system, enter the following command:

```
interlnk d:=
```

## Additional information

If you're handy with hardware, you can make your own parallel cable for use with InterLink. Just be sure that the pins at one end are connected to the pins at the other as follows:

2–15
3–13
4–12
5–10
6–11
15–2
13–3
12–4
10–5
11–6
25–25

The cable set used with LapLink will also work with InterLink. For slower serial port communication, you can purchase a "null-modem" cable at many computer supply stores or Radio Shack.

CROSS REFERENCE

Refer to Appendix D for details about how to connect two computers with InterLink.

*See also* INTERLNK.EXE and INTERSVR.

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

# INTERLNK.EXE

## Description

This device driver is responsible for connecting two computers by way of serial or parallel ports.

## Usage

```
device=[drive:][path]interlnk.exe [/drives:n] [/noprinter]
[/com[:][n or address] [/lpt[:][n or address] [/auto] [/noscan]
[/low] [/baud:rate] [/v]
```

## Options and switches

```
[drive:][path]
```

This option identifies the directory in which INTERLNK.EXE is located, if it is not in the boot directory. By default, SETUP places INTERLNK.EXE in the \DOS directory.

```
[/drives:n]
```

By default, INTERLNK.EXE redirects only three server drives. When the server has drives other than drive C, use this switch to increase the number of supported drives. When you specify a value of zero, only server printers are available.

```
[/noprinter]
```

This switch instructs INTERLNK.EXE not to allow access to the server's printer or printers.

```
[/com[:][n or address]
```

By default, INTERLNK.EXE scans all serial and parallel ports looking for a connection to a remote computer. Use the /COM switch to instruct INTERLNK.EXE to scan only serial ports. INTERLNK.EXE uses a port (without scanning) if you specify it by number (COM1, for example) or by address (3F8).

```
[/lpt[:][n or address]
```

Use the /LPT switch to instruct INTERLNK.EXE to scan only parallel ports. INTERLNK.EXE uses a port (without scanning) if you specify it by number (LPT1, for example) or by address (378).

```
[/auto]
```

Use this switch to instruct INTERLNK.EXE to install only if it can establish an immediate connection with the server. If a server is not found, the device driver is not loaded, and no memory is wasted.

```
[/noscan]
```

This switch instructs INTERLNK.EXE to install without trying to connect to the server.

```
[/low]
```

By default, INTERLNK.EXE tries to load into UMBs. Use this switch to instruct INTERLNK.EXE to load into base memory.

`[/baud:`*rate*`]`

By default, a link is established with a 115200 baud rate. If you experience (intermittent) connection problems, use this switch to slow the data-transmission speed. The following values are accepted: 9600, 19200, 38400, 57600, and 115200.

`[/v]`

Use this switch if you experience timer conflicts when two computers are connected by way of COM ports.

## Remarks

An active link can be established only when the server computer is running INTERSVR.

Because INTERLNK.EXE by default loads automatically into available UMBs, you do not have to load it by using DEVICEHIGH.

If you know which port the client is connected through (and you should if you wired them together), use the /COM*n* or /LPT*n* switch to prevent INTERLNK.EXE from scanning the ports. The system boots faster, and memory is saved. Use MSD to determine the explicit address for the serial and parallel ports.

If you do not plan to use the server's printer, specify /NOPRINTER to save memory.

If the server computer has more than one hard drive, remember to use the /DRIVES switch to enable links with all the server drives.

You cannot execute the following commands executed on a server drive: CHKDSK, DEFRAG, DISKCOMP, DISKCOPY, FDISK, FORMAT, MIRROR, SYS, UNDELETE, and UNFORMAT.

You can interlink two computers using different versions of DOS, but remember that early versions of DOS cannot access drives larger than 32MB.

## Examples

Use the following statement to load INTERLNK.EXE with default settings:

```
device=c:\dos\interlnk.exe
```

Use the following statement in CONFIG.SYS to load the INTERLNK.EXE device driver and establish a link, without printer support, on COM1:

```
device=c:\dos\interlnk.exe /com1 /noprinter
```

Use the following statement to load INTERLNK.EXE and provide access to drives A through E on the server:

```
device=c:\dos\interlnk.exe /drives:5
```

CROSS REFERENCE

### Additional information

Refer to Appendix D for details about how to connect two computers with InterLink.

*See also* INTERLNK and INTERSVR.

| BOB'S SCORECARD 7 | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **INTERSVR** | ■ | ☒ | ☒ | ☐ |

## INTERSVR

### Description

This command establishes a computer as a server in an InterLink session and makes the local drives and printers available to the remote computer.

### Usage

```
intersvr [drive:] [/x=drive:] [/com[:][n or address]] [/lpt[:]
[n or address]] [/baud:rate] [/b] [/v]
```

### Options and switches

`[drive:]`

By default, INTERSVR redirects all drives. Specify one or more individual drives to identify explicitly which drives should be redirected.

`[/x=drive:]`

This switch enables individual drives to remain private and not be redirected. You can specify multiple /X switches.

`[/com[:][n or address]]`

By default, INTERSVR scans all serial and parallel ports to look for a connection to a remote computer. Use the /COM switch to instruct INTERSVR to scan only serial ports. INTERSVR uses a port (without scanning) if you specify it by number (COM1, for example) or by address (3F8).

`[/lpt[:][`*n* or *address*`]]`

Use the /LPT switch to instruct INTERSVR to scan only parallel ports. INTERSVR uses a port (without scanning) if you specify it by number (LPT1, for example) or by address (378).

`[/baud:`*rate*`]`

By default, a link is established with a 115200 baud rate. If you experience (intermittent) connection problems, use this switch to slow the data-transmission speed. The following values are accepted: 9600, 19200, 38400, 57600, and 115200.

`[/b]`

This switch suppresses colors and uses only black and white. Use this option on monochrome or LCD systems.

`[/v]`

Use this switch if you experience timer conflicts when two computers are connected by way of COM ports.

## Remarks

The *server computer* is the computer running INTERSVR, and the *client computer* is the connected computer (using INTERLNK.EXE as a device driver).

You can redirect to the client computer only standard removable drives (such as floppies and Bernoullies) and local hard disks. INTERSVR does not support network drives or CD-ROMs.

You can use the [*drive:*] switch to change the drive letter assignments on the client computer. Drives are assigned on the client computer in ascending alphabetical order and in the same sequence as they are specified on the INTERSVR command line.

When INTERSVR is run from within Windows, DOSSHELL, or any other multitasker, you cannot task switch (you must terminate INTERSVR to return to other tasks).

No special device drivers have to be installed on the computer running INTERSVR.

To exit from INTERSVR, press Alt-F4.

### Examples

Use the following command to start INTERSVR in Automatic Mode, where all drives are redirected:

```
intersvr
```

Use the following command to establish a link on LPT1 and make drive C available to only the client:

```
intersvr c: /lpt1
```

Use the following command to set up an InterLink server, without enabling the floppy drives to be redirected:

```
intersvr /x=a: /x=b:
```

Use the following command to assign drive D on the server as drive D on the client (assuming that the client has only one hard drive) and to assign drive C on the server as E on the client:

```
intersvr d: c:
```

### Additional information

CROSS REFERENCE

Refer to Appendix D for details about how to connect two computers with InterLink.

*See also* INTERLNK and INTERLNK.EXE.

### Extended use

You can use INTERSVR to install the INTERLNK.EXE file on the client system by using the /RCOPY switch as follows:

```
intersvr /rcopy
```

## KEYB

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ■ | ☒ | ☒ |

### Description

This command configures the keyboard for international (non-USA) use.

## Usage

```
keyb [xx[,[yyy] [,[drive:][path]filename.ext]]] [/e] [/id:nnn]
```

Because KEYB is a memory-resident program, INSTALL can load it in CONFIG.SYS by using the following syntax:

```
install=[drive:][path]keyb.com [xx[,[yyy]
[,[drive:][path]filename.ext]]] [/e] [/id:nnn]
```

## Options and switches

When you execute KEYB with no switches, the active keyboard settings are displayed.

`[xx]`

This option specifies a two-character code that represents the country setting.

`[yyy]`

This option identifies the code page you want to use.

`[drive:][path]filename.ext`

This option identifies the keyboard-definition file. KEYB uses KEYBOARD.SYS by default if it is located in the search path.

`[/e]`

Use this switch if you have an Enhanced Keyboard installed on an 8086 system.

`[/id:nnn]`

This switch is used in countries in which more than one keyboard layout is supported (U.K., France, and Italy). This code specifies which keyboard layout to use.

## Remarks

Run HELP KEYB to obtain an up-to-date list of all the supported codes.

KEYB returns the following ERRORLEVELs:

| *ERRORLEVEL* | *Message* |
|---|---|
| 0 | Install successful |
| 1 | Invalid code or syntax used |
| 2 | Keyboard definition file was bad or could not be found |
| 4 | A keyboard or monitor error occurred |
| 5 | Unexpected error |

### Example

Use the following command to configure the keyboard for a French user and to use the alternative character set:

```
keyb fr,437
```

### Additional information

CROSS REFERENCE

Appendix C in *MS-DOS User's Guide* includes layouts for the 20 most common keyboards.

Refer to Chapter 9 in *MS-DOS User's Guide* for a thorough discussion of international configuration issues.

*See also* CHCP, COUNTRY, DISPLAY.SYS, MODE, and NLSFUNC.

## LABEL

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

Every disk has a volume label. Use the LABEL command to change the volume label.

### Usage

```
label [drive:][label]
```

### Options and switches

When you use LABEL with no switches, it displays the current label (and serial number) and prompts for a new label. Press Ctrl-Break to abort.

[*drive:*]

This option specifies the drive whose label is changed. The current drive is selected by default.

[*label*]

This option specifies the new label. The label can be as long as 11 characters and is converted automatically to uppercase.

### Remarks

The label is displayed when you list files with DIR or when you enter the VOL command. Also, the label is used as a security mechanism when you format a hard drive.

LABEL cannot be used on a SUBST drive.

Avoid using the following characters in a label:

`* ? / \ | . , ; : + = [ ] { } & ^ < > “`

### Examples

Use the following command to change the default drive label to *DRIVE C*:

```
label drive c
```

To change the label for drive D to CADDRIVE, enter the following command:

```
label d:caddrive
```

CROSS REFERENCE

### Additional information

*See also* DIR, FORMAT, and VOL.

DISK

### Bob's Shareware Collection disks

CMFiler, from NoVaSoft, is an excellent file manager that provides a myriad of disk- and file-management functions. You can press V to modify a disk volume label.

## LASTDRIVE

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This command specifies how many logical drives DOS recognizes.

### Usage

```
lastdrive=x
```

### Options and switches

*x*

This option specifies a letter in the range from A to Z.

### Remarks

You must use the LASTDRIVE statement in CONFIG.SYS. Try to situate LASTDRIVE before device drivers that consume drive letters.

By default, DOS supports one extra drive in addition to the physical drives.

LASTDRIVE typically is used on systems that connect to networks, use RAM disks, or that use InterLink to connect to another computer's drives.

If you specify too many drives, you waste memory at a rate of approximately 80 bytes per drive. Specify only the drives you need.

### Example

To make sure that DOS supports drives up to the letter M, enter the following command:

```
lastdrive=M
```

### Additional information

Refer to Chapter 5 for a discussion of other memory-saving techniques.

## LH

*See* LOADHIGH.

## LOADFIX

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ■ | ☐ |

### Description

This command corrects the problem of packed file corruption that some programs experience when DOS is loaded HIGH.

### Usage

```
loadfix [drive:][path]programname [program parameters]
```

### Options and switches

*[drive:][path]programname*

This option specifies the name of the program that cannot be run when DOS is loaded HIGH.

*[program parameters]*

This option specifies any parameters required by the problem program.

### Remarks

Some (older) programs do not run if you load them in the first 64K of base memory, and they display a message that the packed file is corrupt. These programs expect DOS to occupy the first 64K of memory and become confused when they are loaded there.

This problem is not a significant one — I have been unable to find even one well-known package that requires this fix. It's there if you need it, however.

### Example

If the program THINGY.EXE displays the message that the packed file is corrupt, try to execute it with LOADFIX:

```
loadfix thingy
```

### Additional information

Refer to Chapter 5 for more information about loading DOS HIGH.

CROSS REFERENCE

*See also* DOS=.

## LOADHIGH

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ■ | ☒ | ☐ |

### Description

In the same way you use DEVICEHIGH to load device drivers in upper memory, you use LOADHIGH to load programs in upper memory.

## Usage

loadhigh [*drive:*][*path*]*programname* [*program parameters*]

## Options and switches

[*drive:*][*path*]*programname*

This option specifies the name of the program that is loaded HIGH.

[*program parameters*]

This option specifies any parameters the program requires.

## Remarks

DOS supports this command on only 386 systems and higher.

You can use DOS memory management software to load programs only if you include HIMEM.SYS, EMM386, and DOS=HIGH statements in CONFIG.SYS, as shown in the following example:

```
device=c:\dos\himem.sys
device=c:\dos\emm386 ram
dos=umb
```

If insufficient free upper memory exists, the program is loaded into conventional memory. To find out whether a program has been loaded HIGH, use the MEM /M command and specify the program name.

The following DOS programs are TSRs that can be loaded HIGH: APPEND, DOSKEY, GRAPHICS, KEYB, MODE, NLSFUNC, PRINT, and SHARE.

The DOS memory optimizer MEMMAKER automatically sets programs (in AUTOEXEC.BAT) to be loaded HIGH when appropriate.

LOADHIGH can be abbreviated as LH.

## Example

Use the following command to load DOSKEY HIGH:

```
loadhigh c:\dos\doskey
```

## Additional information

Refer to Chapter 5 for more information about upper memory.

*See also* DEVICEHIGH, DOS=, EMM386, HIMEM.SYS, and MEM.

## Bob's Shareware Collection disks

UMB_DRV.SYS, from Christopher Blum, supports UMBs on 286 systems and provides advanced memory management for the PCs ranging from 8086 to 486.

## Extended use

By default, LOADHIGH loads the program in the largest free area of upper memory, but this action is not always efficient. To provide more control over memory use, LOADHIGH supports the /L and /S switches also. You normally should leave to MEMMAKER the responsibility of determining the best settings for these switches.

Use the following syntax for these two advanced switches:

```
[[/l:region][,minsize][;region2][,minsize2]...]
```

You use the /L switch to control the region of UMBs in which the program is loaded. (The MEM /F command displays the free regions of upper memory. Device drivers may be too large to fit in one region.) The /L switch enables additional regions to be specified. You can use the MINSIZE parameter to ensure that adequate memory is allocated for the program — some programs require more memory during operation than when they are being initialized.

The following example configuration, thanks to MEMMAKER, uses the /L switch:

```
lh /l:1,6400 c:\dos\\doskey
lh /l:2,13984 c:\dos\share
```

***Note:*** MEMMAKER uses the abbreviated name LH rather than LOADHIGH.

```
[/s]
```

The /S switch (for *S*HRINK) instructs LOADHIGH to reduce the allocated memory after the program has been initialized. You use this switch with programs that need more memory to load than they do during subsequent operation.

# MD

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ■ | ☐ |

## Description

This command creates a new directory.

## Usage

```
md [drive:]path
```

or

```
mkdir [drive:]path
```

## Options and switches

*[drive:]*

This option specifies the drive on which you want the directory to be created. If the drive is omitted, DOS tries to create the directory on the current drive.

*path*

This option specifies the name of the new directory. Do not end the path with a backslash character. If the path begins with "dot dot" (..), the directory is created extending from the parent directory; if the path begins with a backslash (\), the new directory is created from the root directory; otherwise, the new directory is created from the current directory.

## Remarks

DOS can create only one additional directory level at a time. If the directory \THIS did not exist, for example, MD could not, in a single statement, create the directory \THIS\THAT. You first must create the directory \THIS.

You can use the PROMPT and CD commands to identify the current directory.

## Examples

To create the new directory NOTES below the DOS directory, enter the following command:

```
md \dos\notes
```

To create a new directory STUFF on drive A when the default drive is C, enter the following command:

```
md a:\stuff
```

Use the following command to create the directory \123W\BUDGET when the current directory is \123W\SAMPLE:

```
md..\budget
```

## Additional information

*See also* CD, PROMPT, RD, and TREE.

## Bob's Shareware Collection disks

CMFiler, from NoVaSoft, provides an easy interface for managing, moving, and creating directories.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **MEM** | 8 | ■ | ☒ | ☒ | ☐ |

## Description

This command displays valuable information that shows how your system's memory is utilized and how much free memory is available.

## Usage

```
mem [/classify or /debug or /free or /module modulename]
[/page]
```

## Options and switches

When you use MEM with no switches, a 15-line summary of the system's memory utilization is displayed.

```
[/classify]
```

This switch shows all the device drivers and programs loaded in memory, and details the base (conventional) and upper memory use by program and a summary of overall memory use. Because the output exceeds 25 lines on most systems, you should use /P to pause the display after each screen.

```
[/debug]
```

This switch provides detailed information about each device driver and program loaded in memory, in ascending memory order from conventional to upper memory. The starting segment of each program is shown in hexadecimal. Again, you should use the /P switch on most systems.

```
[/free]
```

This switch lists available areas of base and upper memory.

`[/module` *modulename*`]`

This switch provides memory-use statistics about an individual program. Use the /D or /C switch to get the device driver or program name.

```
[/page]
```

You can use this switch with any of the other memory switches to pause after each screen.

## Examples

To display information about each program loaded in memory, enter the following command:

```
mem /c /p
```

To determine how much memory is used by the MSCDEX program, enter the following command:

```
mem /m mscdex
```

To see a summary of free memory, enter the following command:

```
mem /f
```

## Additional information

Chapter 5 shows how MEM is used to help optimize memory use.

The DOS 6 diagnostic program MSD provides a map of upper memory use.

*See also* CHKDSK, DOS=, EMM386.EXE, HIMEM.SYS, and MEMMAKER.

GEM

To check whether a program is already loaded before executing a second copy, you can use the MEM /M progname command. If the program is not loaded already, MEM displays the string *progname* `is not currently in memory.` Although this test is useful from the DOS prompt, it does not help in batch files because MEM does not set an ERRORLEVEL.

The FIND command sets an ERRORLEVEL when a string is not found, however. By piping the output from MEM /C to FIND and testing for the string `not currently`, a batch file easily can test whether a program is loaded in memory.

The following simple batch file uses this technique to test for the DOSKEY, and loads it if it was not found in memory:

```
mem /m doskey | find “not currently”
if errorlevel 1 goto loaded
DOSKEY
goto quit
:loaded
echo DOSKEY was already loaded
:quit
```

You can use this clever trick to test for any program or device driver — yes, even Windows, DesqView, and DOSSHELL. No more need for assembly language programs to determine whether a program is loaded. This technique is explored more fully in Chapter 10.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **MEMMAKER** | 9 | ■ | ☐ | ☒ | ☐ |

DOS 6

## Description

This command assesses your memory configuration and modifies CONFIG.SYS and AUTOEXEC.BAT to make optimum use of memory.

## Usage

```
memmaker [/b] [/batch] [/swap:drive] [/t] [/undo] [/w:n,m]
```

## Options and switches

`[/b]`

The /B switch displays text in black and white. Use this switch on monochrome and LCD systems.

```
[/batch]
```

This switch runs MEMMAKER unattended. Whereas MEMMAKER normally prompts for confirmation of various actions, in Batch Mode MEMMAKER takes the default action without prompting.

`[/swap:`*drive*`]`

This switch advises MEMMAKER of the boot drive letter so that it optimizes the correct CONFIG.SYS and AUTOEXEC.BAT files. This switch is necessary only on systems that swap the boot drive at start-up. Although disk-compression utilities are the usual drive swappers, you do not need to use this switch with DBLSPACE or Stacker 2.0+.

```
[/t]
```

If you experience problems with MEMMAKER and are connected to a token-ring network, use the /T switch to disable token-ring detection.

```
[/undo]
```

Start MEMMAKER with the /UNDO switch to revert to your original system configuration (the configuration before MEMMAKER's last changes).

`[/w:`*n*`,`*m*`]`

Use this switch if you plan to run MEMMAKER in *Express* Mode and want to control how much upper memory is reserved for Windows' translation buffers. The *n* and *m* arguments identify the two buffer sizes in kilobytes. Use a value of /W:0,0 if you don't use Windows. By default, Express Mode uses /W:12,12.

## Remarks

You can use this command only on 386 systems and higher.

MEMMAKER reboots the system. To avoid data loss, close all active applications, including Windows, before running MEMMAKER.

MEMMAKER can be executed in *Express* Mode or *Custom* Mode. In Express Mode, MEMMAKER makes all the decisions; in Custom Mode, you can choose from a variety of optimization options.

## Examples

To use default switches to run MEMMAKER interactively, enter the following command:

```
memmaker
```

To run MEMMAKER and suppress all confirmation prompts, enter the following command:

```
memmaker /batch
```

To revert to the pre-MEMMAKER configuration, enter the following command:

```
memmaker /undo
```

### Additional information

Chapter 5 includes a detailed explanation of how MEMMAKER works and series of tips for how to help MEMMAKER succeed.

*See also* EMM386, HIMEM.SYS, and MEM.

### Bob's Shareware Collection disks

MEMMAKER runs only on 386 systems and higher. For advanced memory support on 286 systems, try UMB_DRV.SYS, from Christopher Blum.

### Extended use

MEMMAKER supports the /SESSION switch, but it is used internally during the optimization process. You should not specify this switch when you execute MEMMAKER.

During the optimization process, MEMMAKER records program and device driver statistics in the MEMMAKER.STS file in the DOS directory. This ASCII file is not deleted when you exit from MEMMAKER. The file contains interesting insights and information about how your programs use memory.

## MENUCOLOR

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This statement enables you to configure the display colors for the multiconfiguration menu.

### Usage

```
menucolor=f[,b]
```

## Options and switches

`f`

This option specifies the menu foreground (text) color. Any value in the range from 0 to 15 is accepted. The default value is 7 (light gray).

`[,b]`

This option specifies the menu background color. Any value in the range from 0 to 15 is accepted. The default value is 0 (black).

## Remarks

The MENUCOLOR statement must be placed in the [MENU] or SUBMENU block of CONFIG.SYS.

The following list shows the 16 color codes:

| **Code** | **Color** | **Code** | **Color** |
|---|---|---|---|
| 0 | black | 8 | dark gray |
| 1 | blue | 9 | light blue |
| 2 | green | 10 | light green |
| 3 | cyan | 11 | light cyan |
| 4 | red | 12 | light red |
| 5 | magenta | 13 | light magenta |
| 6 | brown | 14 | yellow |
| 7 | light gray | 15 | white |

Avoid using colors in the range from 8 to 15 for the background — flashing or blinking text results.

The color of the active topic bar cannot be configured and is always light gray. Avoid using a light gray background (code 7); otherwise, the highlighted bar is not visible.

The color settings apply to the output that AUTOEXEC.BAT commands and other DOS commands generate. Use CLS to clear the display and set the colors back to (boring) light gray on black.

## Examples

Use the following CONFIG.SYS extract to set the menu color to yellow on blue:

```
[menu]
menucolor=14,1
menuitem=Plain,Totally Plain
menuitem=Gainers,Big Gainers
```

Use the following statement to set the foreground color to light red:

```
menucolor=12
```

### Additional information

Refer to Chapter 8 for a thorough discussion of multiconfiguration options.

*See also* MENUDEFAULT and MENUITEM.

## MENUDEFAULT

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This statement sets the default item in a multiconfiguration menu and optionally the number of seconds to wait for user input before applying the default.

### Usage

menudefault=*blockname*[,*timeout*]

### Options and switches

*blockname*

This option specifies the name of the default menu block. This block is highlighted when the menu is first displayed. BLOCKNAME must correspond to a name specified by the MENUITEM statement, but the comparison is not case sensitive.

[,*timeout*]

This option specifies the number of seconds to wait for user input before proceeding with the default block. Any value in the range from 0 to 90 is accepted. A multiconfiguration menu does not have much value if you specify a time-out value of 0: The instant the menu is displayed, the default option is selected, and you have no opportunity to make a selection.

### Remarks

If you choose to use it, the MENUDEFAULT statement must be located in the [MENU] block or a SUBMENU block. If MENUDEFAULT is not specified, DOS sets the default to item 1.

I use a time-out value of nine seconds, which gives me enough time to make a choice, even at 7 a.m. If you change selections by using the cursor keys before the timeout has expired, DOS waits indefinitely for a selection.

### Example

The following CONFIG.SYS extract sets the default selection to the GAINERS block and waits for nine seconds for user input:

```
[menu]
menucolor=14,1
menuitem=Plain,Totally Plain
menuitem=Gainers,Big Gainers
menudefault=Gainers,9
```

### Additional information

Refer to Chapter 8 for a thorough discussion of multiconfiguration options.

*See also* MENUCOLOR and MENUITEM.

## MENUITEM

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

This statement identifies a menu option on a multiconfiguration menu.

### Usage

`menuitem=`*blockname*`[`, *item description*`]`

### Options and switches

*blockname*

This option specifies the name of the block of configuration commands to be executed, if the item is selected. Elsewhere in the file, a unique block heading must have the format [*blockname*]. *blockname* is not case sensitive.

[ , *item description*]

By default, the item description is the blockname. You can optionally specify an item description, which can include spaces and is displayed in the same case in which it is entered. The text can contain as many as 70 characters.

## Remarks

Do not assign a blockname of *menu* or *common* because these blocknames are reserved for other uses. Blocknames may not include the characters \ / , ; = [ ] or the space character.

Every item in the menu automatically has a number prefix, and you can have up to nine menu items in each menu.

In addition to processing the commands located in the [*blockname*] block, DOS processes commands located in any blocks with the name *[common]*.

If DOS can't find a block with the specific blockname, the item will not appear in your menu.

## Examples

The following CONFIG.SYS extract defines a menu on which the blockname is used as the description:

```
[menu]
menucolor=14,1
menuitem=Frugal
menuitem=Everything
menudefault=Everything,9
.....
[frugal]
.....
[everything]
.....
```

A topic description has been added to the following similar menu:

```
[menu]
menucolor=14,1
menuitem=Frugal,Max memory but no CD or network
menuitem=Everything,All optional drivers loaded
menudefault=Everything,9
.....
[frugal]
.....
[everything]
.....
```

### Additional information

You should use the SUBMENU command if the item needs to display a secondary configuration menu.

Refer to Chapter 8 for more information on MENUITEM.

*See also* MENUCOLOR, MENUDEFAULT, and SUBMENU.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **MIRROR** | 8 | ■ | ■ | ■ | ☐ |

***Note:*** MIRROR is not officially part of DOS 6. The DOS 5 version of MIRROR is, however, included on the DOS 6 Supplemental programs disk, available from Microsoft. If you have upgraded from DOS 5, you can use the copy of MIRROR that came with it — it should still be in your DOS directory.

### Description

The deletion tracking feature provided by MIRROR (using the /T switch) is now provided by UNDELETE, so is not documented here. However, every DOS 6 user should use MIRROR to save a copy of the FAT and root directory for each hard disk. In addition, MIRROR should be used with the /PARTN switch to save a copy of the hard disk partition table.

### Usage

```
mirror [drive:[....]] [/1]
```

or

```
mirror /partn
```

### Options and switches

When you use MIRROR without any switches, the FAT and root directory details are saved for the default drive.

```
drive:[...]
```

This option identifies one or more drives whose FAT and root directory details will be saved in hidden files.

```
[/1]
```

By default, if MIRROR sees an existing copy of MIRROR.FIL (the hidden file containing all the FAT and root directory data), it will copy the file to MIRROR.BAK, before creating a new MIRROR.FIL file. The /1 switch instructs MIRROR not to keep a backup copy, that is, to keep only one file.

```
/partn
```

This switch instructs MIRROR to make a copy of the hard disk partition table and save it to a diskette.

## Remarks

The MIRROR command creates vital data recovery information and should be used in AUTOEXEC.BAT to make copies of the partition table and copies of the FAT and root directory for each hard drive.

If the /1 switch is used, MIRROR will delete the file MIRROR.BAK if it finds one.

You can determine whether a disk contains the hidden MIRROR files by using the UNFORMAT command with the /J switch and pressing Esc when prompted.

## Examples

Use the following command to create the hidden MIRROR files that record the FAT and root directory of the default drive:

```
mirror
```

To make a copy of the FAT and root directory for drives C, D, and E, enter the following command:

```
mirror c: d: e:
```

To make a copy of the hard disk partition table, enter the following command:

```
mirror /partn
```

## Additional information

Refer to Chapter 3 for a full explanation of how the MIRROR files are used during an UNFORMAT operation.

*See also* FORMAT, UNDELETE, and UNFORMAT.

## MKDIR

*See* MD.

## MODE

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

### Description

For clarity, the MODE command is divided into the following subcommands:

- `mode comn`
- `mode device CP`
- `mode device /Status`
- `mode display`
- `mode lptn`
- `mode con`

## MODE COM*n*

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

### Description

This command sets the communication settings for a serial port.

### Usage

```
mode comn[:] [b,[p[,d[,s[,r]]]]]
```

or (in longhand)

```
mode comn[:] [baud=b] [parity=p] [data=d] [stop=s] [retry=r]
```

## Options and switches

`comn[:]`

This option specifies the serial port whose settings are adjusted. MODE supports four serial ports (COM1, COM2, COM3, and COM4).

`[baud=b]`

This option specifies the port's data-transmission rate, where *b* is the first two digits of the baud rate. The following table lists the supported rates.

| ***Code*** | ***Baud rate*** |
|---|---|
| 11 | 110 |
| 15 | 150 |
| 30 | 300 |
| 60 | 600 |
| 12 | 1,200 |
| 24 | 2,400 |
| 48 | 4,800 |
| 96 | 9,600 |
| 19 | 19,200 |

`[parity=p]`

This option specifies the method used for parity checking. *p* can be one of the following letters: N (none), E (even), O (odd), M (mark), or S (space). Parity checking is set to *even* by default.

`[data=d]`

This option indicates how many bits comprise each character. The value is 7 by default, but a value in the range from 5 to 8 is accepted.

`[stop=s]`

This option indicates how many bits define the end of each character. The value is 2 by default (or 1, at the slowest rate of 110 baud), and the values 1, 1.5, and 2 are accepted.

```
[retry=r]
```

This option specifies how transmission errors should be handled. *r* can be one of the following characters:

| *Letter* | *Action* |
|---|---|
| E | Returns an error when the port is busy |
| B | Returns the busy code when the port is busy |
| P | Keeps retransmitting until the printer accepts the data |
| R | Returns the ready code when the port is busy |
| N | Ignores a busy port and takes no action |

## Remarks

You can omit parameters by using the abbreviated syntax, but you must use commas as placeholders to indicate which parameters are null.

If any of the parameters are omitted, the preceding setting for the specified serial port is used.

## Examples

Use the following command to set the COM2 port communication parameters:

```
mode com2: baud=2400 parity=n data=8 stop=1 retry=p
```

You can execute the same command in abbreviated form as follows:

```
mode com2:2400,n,8,1,p
```

You can later modify the same port to not retry with the following command:

```
mode com2:,,,,n
```

# MODE CON

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

## Description

This command adjusts the keyboard's typematic rate.

## Usage

```
mode con[:] [rate=r delay=d]
```

### Options and switches

`[rate=r]`

This option specifies how quickly additional characters are entered when a key is held down. The rate approximates to characters per second, and a value in the range from 1 to 32 is accepted. The default is 20.

`[delay=d]`

This option sets the amount of time a key must be held down before additional characters are automatically entered. The following values are accepted: 1 (0.25 second), 2 (0.5 second), 3 (0.75 second), and 4 (1 second).

### Remarks

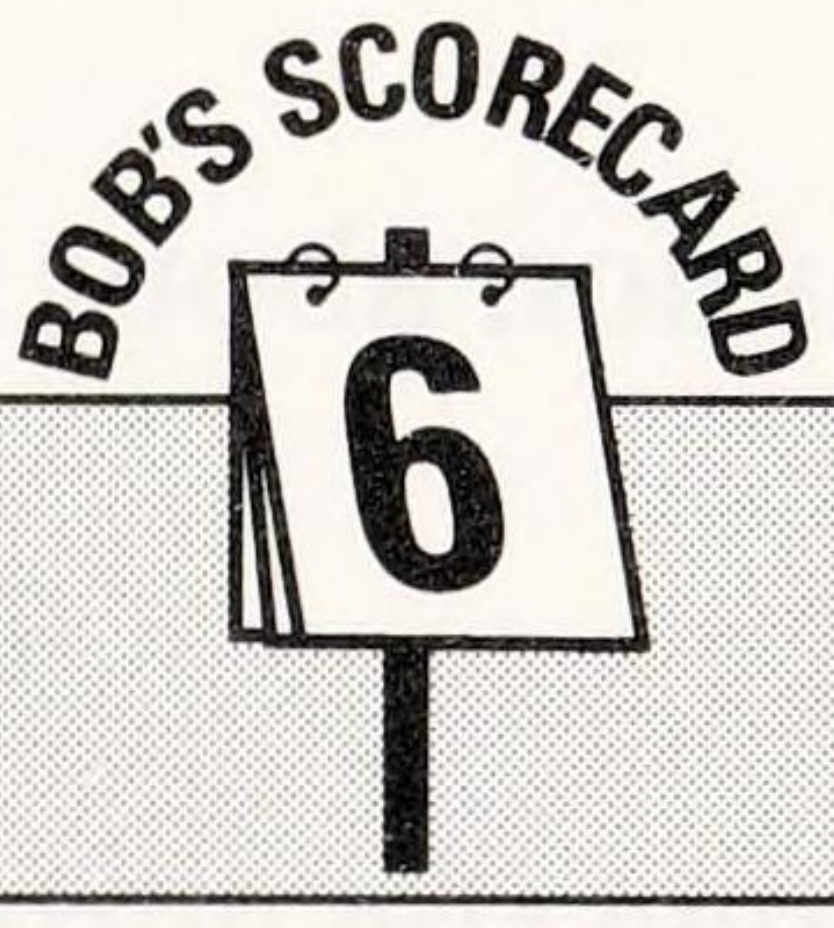
Some older systems do not allow the typematic rate to be adjusted.

### Example

Use the following command to set the keyboard at full throttle:

```
mode con: rate=32 delay=1
```

## MODE *device* CP



| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

### Description

This command manages the code pages for the monitor, keyboard, and printer.

### Usage

The MODE *DEVICE* CP command has the following four primary formats:

```
mode device cp prepare=((yyy [...]) [drive:][path][filename.ext])

mode device cp select=yyy

mode device cp refresh

mode device cp /status
```

## Options and switches

*device*

This option specifies the console or printer device. The following devices are supported: CON, LPT1, LPT2, and LPT3.

`prepare ((`*yyy* `[...]) [`*drive:*`][`*path*`][`*filename.ext*`])`

A code page must be prepared before it can be selected. *yyy* identifies the appropriate code page. Execute HELP MODE to display the list of supported code pages. Optionally, specify the name of the code-page information file — these files are installed in the DOS directory and have an extension of CPI.

`select=`*yyy*

After a code page is prepared, you can select it with the SELECT switch.

`refresh`

This option re-installs a code page after it has been hosed by a rogue program.

`/status`

This switch displays information about the installed code pages.

## Remarks

In general, use CHCP to change the code page in preference to the MODE command. CHCP changes all devices in one command and ensures that the country and code-page settings match.

The following table details the DOS 6 code-page information files located in the DOS directory.

| *File* | *Description* |
|---|---|
| EGA.CPI | For EGA displays on PS/2s |
| 4201.CPI | IBM Proprinter models 4201 and 4202 |
| 4208.CPI | IBM Proprinter models 4207 and 4208 |
| 5202.CPI | IBM Quietwriter III |
| LCD.CPI | IBM PC Convertible liquid crystal display |

### Example

Use the following commands with MODE to prepare the code page and CHCP to activate the code page:

```
mode con cp prepare=((863)c:\dos\ega.cpi)
chcp 863
```

### Additional information

CROSS REFERENCE

*See also* CHCP and NLSFUNC.

## MODE *device* /STATUS

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

### Description

This command instructs MODE to display information about an individual device or all devices.

### Usage

```
mode [device] /status
```

### Options and switches

[*device*]

This option specifies the name of a device (COM1, COM2, COM3, COM4, CON, LPT1, LPT2, LPT3, or PRN). If no device is specified, the status of all devices is displayed.

### Remarks

Use the | MORE pipe to pause the display after each screen.

### Examples

Use the following command to display the status of all devices:

```
mode /status | more
```

Use the following command to display the status of LPT1:

```
mode lpt1 /status
```

## MODE display

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

### Description

This command sets the active display-adapter configuration.

### Usage

The MODE DISPLAY subcommand has the following three formats:

```
mode [display] [,shift[,t]]
```

```
mode [display] [,n]
```

```
mode con[:] [cols=c] [lines=n]
```

### Options and switches

```
[display] [,shift[,t]]
```

*display* can have one of the following values:

| *Value* | *Description* |
|---|---|
| 40 or 80 | Sets the number of characters across the screen (columns) |
| bw40 or bw80 | Sets the number of characters and forces the display mode to black and white |
| co40 or co80 | Sets the number of characters and forces the display mode to color |
| mono | Sets the number of characters to 80 on a monochrome display adapter |

*shift* shifts the display to the left or right on CGA systems. Accepted values are L (left) and R (right).

```
t
```

This option displays a test pattern on CGA systems.

```
con[:] [cols=c] [lines=n]
```

This option sets the number of characters per line (`cols`) and lines per screen (`lines`). *c* can have a value of 40 or 80. *n* can be 25, 43 (on EGA and VGA systems), or 50 (on VGA systems). You can use this MODE command format only on systems on which ANSI.SYS is installed.

## Remarks

One of my favorite uses for the MODE co80 command is to restore the cursor when a program has crashed and the cursor has disappeared. You can use the same command to return the system to Text Mode when a graphics program crashes and drops you to DOS in Graphics Mode.

If you have vision problems or have difficulty reading the display, you may benefit from using the MODE co40 command to switch to 40-character mode.

Many contemporary programs (especially full-screen applications) ignore the active mode setting.

## Examples

To switch to black-and-white mode with 40 characters per line, enter the following command:

```
mode bw40
```

If ANSI.SYS is installed, you can enter the following command on a VGA system to switch to 50-line mode:

```
mode con: cols=80 lines=50
```

# MODE LPT*n*

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

## Description

This command redirects printer output from a parallel port to a serial port or reconfigures a parallel port.

## Usage

```
mode lptn[:]=comn[:]
```

## Options and switches

```
lptn[:]
```

This option specifies the parallel port whose output is redirected to the serial port. Valid values are LPT1, LPT2, and LPT3.

```
comn[:]
```

This option specifies the serial port that receives the output sent to the parallel port. Valid values are COM1, COM2, COM3, and COM4.

## Remarks

In most cases, you must use MODE `com`*n* to configure the serial port before redirecting an LPT port to it.

## Example

Use the following command to set COM2 to 9600 baud with even parity, and then redirect LPT2 output to it:

```
mode com1 96,e,,,p
mode lpt2=com2
```

## Extended use

The MODE `lpt`*n* command also is used, although rarely, to reconfigure a parallel port, by using the following syntax:

```
mode lptn[:] [c][,[l][,r]]
```

or by using the following longhand syntax:

```
mode lptn[:] [cols=c] [lines=l] [retry=r]
```

*c* specifies the number of characters per line and can be either 80 or 132. *l* specifies the number of lines per inch and can be 6 or 8. *r* specifies the retry setting and can have one of the following values:

| *Value* | *Description* |
|---|---|
| e | Returns "error" on a busy port |
| p | Keeps retrying until the printer accepts the character |
| r | Returns `ready` on busy port |
| n | No retry action (default) |

# MORE

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 7 | ■ | ☒ | ☒ | ☐ |

## Description

This command pauses the output from a command-line utility after every screen (it stops text from scrolling off the top of the screen).

## Usage

*command* [*command–parameters]* | `more`

or (to display a file's contents)

`more <` [*drive:*][*path*]*filename.ext*

## Options and switches

*command*

This option specifies any command-line program whose output might fill the screen.

[*command–parameters*]

Optional parameters required by the command-line program.

[*drive:*][*path*]*filename.ext*

This option specifies the name of an ASCII file whose contents you want to view.

## Remarks

Any lines longer than the display width (usually 80 but can be 40) are wrapped to the next line.

When the screen has been filled, MORE displays this string:

```
– MORE –
```

Press any standard key to see the next screen. MORE does not pause the output for the last partial screen.

The commands DIR and MEM provide the /P switch to pause the display after each screen to avoid the need to use MORE with them.

## Examples

To display the status of all devices and pause after each screen, enter the following command:

```
mode /status | more
```

To view the contents of the file READ.ME, enter the following command:

```
more < read.me
```

### Additional information

MORE is used often with the following DOS commands: ATTRIB, MODE /STATUS, TREE, and TYPE.

### Bob's Shareware Collection disks

The outstanding LIST program, from Buerg Software, is the champion of file viewers. You can page up *and* page back through a file, search for text, view in hexadecimal, display a ruler, leave, and jump to bookmarks.

CMFILER, the first-class file manager from NoVaSoft, has the capability to view files.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **MOVE** | 7 | ■ | ☒ | ☒ | ☐ |

### Description

This command moves files from one path to another and renames directories.

### Usage

`move [`*drive:*`][`*path*`]`*filemask*`[, [[`*drive:*`][`*path*`]`*filemask*`[...]]` *target*

### Options and switches

[*drive:*][*path*]*filemask*

This option identifies the name of the source file to be moved. Wildcards are not supported. This option can also be a directory name if the intent of the MOVE operation is to rename a directory; you must specify this switch.

[, [[*drive:*][*path*]*filemask*[...]]

You can specify additional source files, with each filename being separated by a comma.

*target*

This option identifies the target location where the files will be copied. The target location can be a drive letter followed by a colon, a directory name, or a combination of the two. If a single source file or directory is specified, to rename the file or directory you can specify a new filename or directory as the target.

## Remarks

Unlike COPY, the MOVE command requires both a source and target to be specified on the command line.

If you're moving a file to a new path on the same drive, MOVE will just modify the file's directory entries to relocate it in the new path; the file's data remains in the same physical location on the disk. This process is much faster than actually moving the file's data.

When using MOVE to rename directories, the source and target directories must both stem from the same parent directory — you cannot use MOVE to restructure the directory hierarchy.

## Examples

Use the following command to move all the files with the WK1 extension from the current directory to the directory C:\OLDFILES:

```
move *.wk1 c:\oldfiles
```

To move the files HIGH.DBF and LOW.DBF from the current directory to the D:\DBASE\PLAN directory, enter the following command:

```
move high.dbf,low.dbf d:\dbase\plan
```

Use the following command to move the file WELCOME.DOC from the directory C:\AMIPRO\DOCS to the directory C:\AMIPRO\SEMINAR and rename the file HOWDY.DOC:

```
move c:\amipro\docs\welcome.doc c:\amipro\seminar\howdy.doc
```

To rename the directory D:\DBASE\FILES\BUDGET to D:\DBASE\FILES\FORECAST, enter the following command:

```
move d:\dbase\files\budget d:\dbase\files\forecast
```

CROSS REFERENCE

## Additional Information

Refer to Chapter 2 for a discussion of MOVE.

*See also* COPY, XCOPY.

DISK

## Bob's Shareware Collection disks

PocketD, from Pocketware, can move files with far more flexibility than DOS can, and it includes the following features: multiple wildcards are supported;

files can be selected based on file size, date stamp, and even contents; optional prompts before overwriting files, and much more. PCOPY from Patri-Soft and CMFILER from NoVaSoft also provide excellent file moving options.

|  | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **MSAV** | 9 | ■ | ☒ | ■ | ☐ |

## Description

This command runs the full-screen utility Microsoft Anti-Virus, which scans drives for viruses and optionally removes them.

## Usage

```
msav [drive: or /a or /l] [/s or /c] [/r] [/n or /p [/f]]
[/video] [/videomode]
```

## Options and switches

When you run MSAV with no switches, it displays a full-screen menu.

`[drive:]`

This option specifies the drive to scan for viruses. If no drive is specified, the default drive is scanned.

`[/a]`

Use /A to scan all drives except floppies.

`[/l]`

Use /L to scan all drives except network drives.

`[/s]`

The /S switch scans for viruses but does not correct them.

`[/c]`

The /C switch scans and cleans (removes) viruses.

```
[/r]
```

This switch instructs MSAV to create a detailed report in the ASCII file MSAV.RPT in the root directory of the scanned drive.

```
[/n]
```

This switch runs MSAV in command-line mode (rather than the default full-screen mode) and automatically creates the MSAV.RPT file in the root directory. The name of the active file is displayed during the scanning procedure.

```
[/p]
```

The /P switch runs MSAV in command-line mode also. This switch is similar to /N, but the messages are displayed on-screen rather than written to a report file.

```
[/f]
```

You can use this switch in conjunction with /N or /P to suppress the display of filenames during the scanning procedure.

```
[/video]
```

This switch instructs MSAV to display a list of all the optional command-line switches used to control the video display characteristics.

[/*videomode*]

This switch sets the display characteristics in full-screen mode. Run MSAV /VIDEO to see a list of valid switches.

## Remarks

By default, MSAV creates a checksum file named CHKLIST.MS in every directory of the scanned drive. This file is used to record a unique value for every file. During subsequent virus checks, MSAV recomputes the checksum and compares it to the value stored in CHKLIST.MS; if the checksum value is different, the file has been modified.

The CHKLIST.MS file provides an added level of security. If you don't want hundreds of CHKLIST.MS files scattered throughout your system, however, select **O**ptions from the MSAV main menu and uncheck the Create Ne**w** Checksums option.

TIP

If you have installed the Windows version, MWAV, you can delete all the CHKLIST.MS files by selecting the menu option Scan Delete CHKLIST Files. Alternatively, use the PocketD shareware program to delete them in one command.

All the optional MSAV settings are stored in the ASCII file MSAV.INI in the DOS directory. You can instruct MSAV to access the MSAV.INI file in a specific directory by setting the environment variable MSDOSDATA equal to the specific directory in which the active INI file is located (SET MSDOSDATA=D:\VIRUS, for example).

### Examples

To start MSAV in interactive full-screen mode, enter the following command:

```
msav
```

Enter the following command to scan but not remove viruses on all drives except floppies:

```
msav /s /a
```

### Additional information

CROSS REFERENCE

Refer to Chapter 3 for a thorough discussion of viruses.

If you have installed the Windows utilities for DOS 6, you may prefer to use MWAV, the Windows version of MSAV. It provides all the same functionality but takes full advantage of Windows' rich graphical environment.

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 9 | ■ | ☒ | ■ | ☐ |

## MSBACKUP

DOS 6

### Description

This command runs the full-screen utility Microsoft Backup, which backs up and restores copies of your files.

### Usage

`msbackup [`*setup filename*`] [/bw` or `/lcd` or `/mda]`

### Options and switches

When MSBACKUP is run with no switches, a full-screen menu is displayed.

`[`*setup filename*`]`

This option specifies the name of an MSBACKUP setup file (with the extension SET) that details the files to be backed up and the configuration. If you do not specify a setup file, DEFAULT.SET is used.

`[/bw]`

This switch instructs MSBACKUP to suppress colors and use black and white.

`[/lcd]`

This switch instructs MSBACKUP to use a color scheme suitable for LCD laptop displays.

`[/mda]`

Use this switch on monochrome systems.

## Remarks

For the first time in the history of DOS, a useful file-backup utility is included with the operating system. Good-bye, BACKUP — hello, MSBACKUP.

MSBACKUP is a full-screen backup utility for saving *and* restoring files. It can perform full backups, incremental backups, backups based on date, or backups of only selected files.

You control the backup settings with a backup SET file. Use the File menu to save the current configuration in a SET file or to load a configuration from a SET file. MSBACKUP searches for SET files in three locations:

1. The environment, to see whether a path is specified for the MSDOSDATA variable
2. The MSBACKUP directory
3. The default directory

MSBACKUP contains a wealth of on-line help. Press F1 at the menu to browse through the help text.

You can use the old-style RESTORE program to restore files backed up with old versions of BACKUP, but RESTORE is *not* used with MSBACKUP data. MSBACKUP uses a special catalog file when it restores files. One copy of the catalog is stored on the last disk of the backup, and another is stored by default in the DOS directory. Use the Backup Set Catalog list box to select a catalog. If you cannot find the backup catalog, you can instruct MSBACKUP to rebuild the catalog by selecting the Catalo**g** option from the **R**estore menu.

MSBACKUP does not back up to tape, but you can back up to any disk drive, hard disk, or network drive.

On systems on which Windows 3.0 or higher is installed, SETUP adds the file VFINTD.386 to the system directory. This file enables MSBACKUP to be run in a DOS session within Windows in Enhanced Mode. If you are using Windows, however, use MWBACKUP rather than MSBACKUP.

MSBACKUP provides no switches for automating the backup execution — you must run the full-screen application and make menu selections.

MSBACKUP must be started from a hard disk.

### Examples

Use the following command to run MSBACKUP and present the main menu:

```
msbackup
```

If you have a backup SET file called BUDGET.SET, you can enter the following command to load the configuration file automatically:

```
msbackup budget.set
```

### Additional information

CROSS REFERENCE

MWBACKUP, the Windows version of MSBACKUP, provides the same features and capabilities. Refer to Chapter 3 for a full discussion of the backup tools.

GEM

If MSBACKUP reports the error that the DMA buffer size is too small, use the D=96 switch with EMM386.EXE to increase the DMA buffer size, and try again.

## MSCDEX

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

DOS 6

### Description

MSCDEX (short for *M*icro*s*oft *CD*-ROM *Ex*tensions) makes a CD-ROM behave like a standard disk device and assigns a logical drive letter to the CD-ROM drive. Most CD-ROM drives can only be accessed after MSCDEX has been installed.

### Usage

```
mscdex [/e] [/k] [/s] [/v] [/d:name[...]] [/l:drive]
[/m:buffers]
```

## Options and switches

`[/e]`

This switch instructs MSCDEX to store the CD-ROM cache buffers in expanded memory.

`[/k]`

Use this switch when accessing CD-ROM disks that contain files stored using the Japanese Kanji format. It's not a switch I use everyday.

`[/s]`

Short for *share,* /S allows the CD-ROM to be accessed by other network users, provided you are running network software like Windows For Workgroups.

`[/v]`

Short for *verbose,* /V instructs MSCDEX to display memory usage statistics when it loads.

`[/d:`*name*`[...]]`

This switch specifies the name of the CD-ROM device driver loaded in CONFIG.SYS. Normally, you would only specify a single /D switch, but if you have multiple CD-ROM drives, you should specify each device's driver name.

`[/l:`*drive*`]`

This switch specifies the drive letter that will be assigned to the drive. By default, MSCDEX assigns the next available drive letter.

`[/m:`*buffers*`]`

This switch specifies the number of sectors to cache. Each sector consumes a little under 2K of memory, and the default value is 10 sectors. There are only marginal performance advantages in using a higher value.

## Remarks

MSCDEX can be loaded high with the LOADHIGH (or LH) command to save base memory.

MSCDEX will only load successfully if a CD-ROM device driver (supplied by the hardware vendor) is also loaded in CONFIG.SYS. Most CD-ROM device drivers support a `/d:`*name* switch to assign the device driver a unique name. The device driver name specified in CONFIG.SYS must correspond exactly with the name specified using MSCDEX's /D switch. If a matching device driver was not successfully loaded, MSCDEX will fail and display the message

```
No valid CD-ROM device driver selected.
```

***Note:*** CONFIG.SYS may include two device drivers that appear related to the CD-ROM. The second device driver provides the sound support.

To conserve memory, configure EMM386 to support expanded memory and specify the /E switch to instruct MSCDEX to install the CD-ROM buffers in expanded memory. Using default settings, MSCDEX will consume 35K of memory (high or base), but when /E is specified, the memory usage drops to 16K.

## Examples

If a CD-ROM device driver was installed in CONFIG.SYS using the /D:MSCD001 switch, use the following command to install MSCDEX, using default settings:

```
mscdex /d:mscd001
```

Use the following command to install MSCDEX with 15 expanded memory sector buffers and display verbose installation messages:

```
mscdex /d:mscd001 /m:15 /e /v
```

Use the following command to make the CD-ROM drive accessible from other workstations on the network and install the buffers in expanded memory:

```
mscdex /d:mscd001 /s /e
```

## Additional information

CROSS REFERENCE

*See also DE*VICE, EMM386, LOADHIGH, MEMMAKER.

BOB'S SCORECARD 9

# MSD

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ■ | ☐ |

## Description

This command analyzes and reports on your detailed system configuration.

## Usage

MSD is used interactively with the following syntax:

```
msd [/i] [/b]
```

or MSD can create reports by using the following syntax:

msd [/i] [/f[*drive:*][*path*]*filename.ext*] [/p[*drive:*][*path*]*filename.ext*] [/s[*drive:*][*path*][*filename.ext*]]

## Options and switches

[/i]

This switch instructs MSD not to scan the hardware. Use this option when MSD cannot load and hangs while it is examining your system.

[/b]

This switch instructs MSD to suppress colors and use black and white.

[/f [*drive:*][*path*]*filename.ext*]

This switch instructs MSD to create an ASCII file that reports your system's details. Before the report is created, MSD prompts for your name, address, and a one-line comment. These details are written at the top of the report.

[/p [*drive:*][*path*]*filename.ext*]

Like /F, the /P switch instructs MSD to record its findings in a file, but it doesn't prompt for name and address details.

[/s [*drive:*][*path*][*filename.ext*]]

This switch writes a summary of your system's configuration. The output is written to the display by default, but you can save it in a file by specifying a filename.

## Remarks

MSD was first released (but not documented) with Microsoft Word 5.5, and later an enhanced version was released with Windows 3.1. It proved to be such a valuable tool that Microsoft enhanced it still further and included it in DOS 6.

MSD is designed as a diagnostics tool because it provides detailed information about your system's configuration. On one of my systems, it generates more than 40 pages of statistics.

The /F*filename* switch is well suited to those of us involved in telephone technical support — you can instruct a remote user to run MSD with the /F switch, enter details, and then send the file to you. The file contains your clients' details and information about the following topics:

- Computer and BIOS details
- Memory
- Video
- Network
- Operating system
- Environment settings
- Mouse
- Game adapters
- Disk drives
- LPT ports
- COM ports
- IRQ settings
- TSR programs
- Device drivers

It even includes copies of the computer's key configuration files, including CONFIG.SYS, AUTOEXEC.BAT, SYSTEM.INI, WIN.INI, PROTOCOL.INI, and DBLSPACE.INI.

When you run MSD interactively, it first snoops around your hardware to assess the system configuration. On some systems, this process can take as long as a couple of minutes. Be patient.

If MSD does not load, use the /I switch and select each category individually. Eventually, you select the category that is causing the problem, and the system hangs. You then know the area of your system to investigate.

### Examples

To run MSD interactively and select information from a menu, enter the following command:

```
msd
```

To prompt for name and address details and then write your computer details to the file MSD.OUT, enter the following command:

```
msd /f msd.out
```

### Additional information

Refer to Chapter 2 for more information on MSD.

*See also* CHKDSK, MODE /STATUS, and MEM.

## MWAV

BOB'S SCORECARD 9

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ☐ |

### Description

This command is the Windows version of MSAV. Refer to the MSAV section for more information.

## MWAVTSR

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ☐ |

### Description

This command provides Windows support for messages generated by VSAFE. You should load it from Windows when VSAFE is installed. See VSAFE for more information.

# MWBACKUP

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 9 | ☐ | ☐ | ☐ | ☐ |

## Description

This command is the Windows version of MSBACKUP.

The first time you run MWBACKUP, it invokes an automatic test procedure. This procedure involves checking the DMA integrity and performs a small backup-and-restore operation to ensure that everything is functioning reliably. Have two disks ready for the small backup — use the disk drive you will select for real backups.

MWBACKUP and MSBACKUP are compatible. See MSBACKUP for more information.

GEM

If MWBACKUP reports the error that the DMA buffer size is too small, add a SETDMABUFFERS=96 entry in SYSTEM.INI to increase the DMA buffer size.

# MWUNDEL

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 9 | ☐ | ☐ | ☐ | ☐ |

DOS 6

## Description

You use this Windows version of UNDELETE to recover accidentally deleted files and directories.

## Usage

```
mwundel
```

## Options and switches

None.

## Remarks

When you select the DOS utilities for Windows during SETUP, a Microsoft Tools group is created automatically. Double-click the Undelete icon to run MWUNDEL.

Both UNDELETE and MWUNDEL can perform three types of file undeletion: sentry, tracker, and DOS. Refer to the UNDELETE section for more details.

In addition to the features described in UNDELETE, MWUNDEL can search an entire drive for a deleted file.

When a directory is selected, MWUNDEL automatically displays a list of all the deleted files and a statement of the files' condition. This condition gives an indication of the chances for successful undeletion and has one of the following values:

| ***Value*** | ***Message*** |
|---|---|
| Perfect | The file can be undeleted automatically. |
| Excellent | Because all the files clusters are consecutive and none of them have been used, in all likelihood the file can be recovered successfully. |
| Good | It appears that all the file's clusters are available, although they are not contiguous. You should be able to undelete the file. |
| Poor | The first file cluster is being used by another file. The chances of recovery are slim to none. |
| Destroyed | All the file's clusters have been reused. |
| Recovered | You have undeleted the file in this session. |

MWUNDEL cannot undelete files categorized as poor or destroyed. Try the DOS command UNDELETE on these files.

CROSS REFERENCE

## Additional information

Chapter 3 explains a number of ways to guard against data loss.

*See also* UNDELETE.

BOB'S SCORECARD 6

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **NLSFUNC** | ☒ | ■ | ☒ | ☒ |

DOS 6

## Description

This command provides *N*ational *L*anguage *S*upport by enabling code-page switching.

## Usage

From the command line, NLSFUNC has the following syntax:

```
nlsfunc [[drive:][path]filename.ext]
```

Because NLSFUNC is a TSR, you can install it from CONFIG.SYS by using the INSTALL command as follows:

```
install=[idrive:][path]nlsfunc.exe [[drive:][path]filename.ext]
```

## Options and switches

`[[drive:][path]filename.ext]`

This option specifies the name of the country driver. DOS provides the COUNTRY.SYS driver for this function. Alternatively, you can use the COUNTRY= directive before the NLSFUNC entry to identify the location and name of the country driver.

## Remarks

You should be concerned about this command only if you are configuring DOS for international use.

On 386 systems and higher, you can use the LOADHIGH directive in AUTOEXEC.BAT to load NLSFUNC into UMBs. NLSFUNC uses about 3K of memory.

## Example

Use the following statement from AUTOEXEC.BAT to install national language support:

```
nlsfunc
```

## Additional information

Refer to Chapter 9 of the *MS-DOS User's Guide* for a detailed description of how DOS can be configured for international use.

*See also* CHCP, COUNTRY, DISPLAY.SYS, KEYB, and MODE CP.

| BOB'S SCORECARD 6 | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **NUMLOCK** | ☐ | ☐ | ☐ | ■ |

## Description

This command controls whether the pesky keyboard Num Lock key is set to On or Off at system boot-up.

## Usage

```
numlock=[on or off]
```

## Options and switches

```
[on or off]
```

It should come as no surprise that you specify ON to turn Num Lock on and OFF to turn it off.

## Remarks

The NUMLOCK statement must be located in the `[MENU]` section of CONFIG.SYS.

## Example

The following CONFIG.SYS extract declares a menu and turns off Num Lock:

```
[menu]
menucolor=14,1
menuitem=Plain,Totally Plain
menuitem=Gainers,Big Gainers
menudefault=Gainers,9
numlock=off
```

## Additional information

Chapter 8 explores system-configuration options in detail.

*See also* MENUCOLOR, MENUDEFAULT, MENUITEM, and SUBMENU.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **PATH** | 7 | ☒ | ■ | ■ | ☐ |

## Description

This command stores a list of directories that DOS searches when it looks for a program or batch file.

## Usage

```
path [[drive:][path][;...]]
```

## Options and switches

When you execute the PATH command with no switches, the active search path is displayed.

```
[drive:][path][;...]
```

This option specifies a list of directories separated by colons. Normally, you should include the drive so that DOS searches the correct path no matter which drive is current.

## Remarks

Every computer with a hard disk should have a PATH statement included in AUTOEXEC.BAT.

The search path is stored in the DOS environment as the PATH variable. Enter the SET command, with no parameters, to see the complete environment.

When you execute a command, DOS determines whether it is an internal DOS command and, if not, searches the current directory. DOS searches every directory specified in the path in the order in which it is specified. As soon as it finds a program, the search terminates and the program is executed.

To improve performance, locate the most frequently accessed directories toward the beginning of the path.

Do not add too many directories to the path — DOS slows down as it searches all the specified directories for a command. Furthermore, avoid putting floppy disk directories in the search path. DOS really slows down as it accesses the floppies to look for a program.

Despite the pleas from users, the PATH is still limited to a maximum of 127 characters. To fit more directory names into the PATH, you need to resort to tricks such as shortening your directory names, using the SUBST command to substitute the shorter logical names for your longer directory names, or using the `append /x:on` command. Chapter 2 explains some other techniques that may alleviate PATH problems.

### Examples

Use the following PATH statement to instruct DOS to search four directories for programs:

```
path c:\windows;c:\dos;c:\;c:\batfiles
```

Use the following batch file statement with the environment variable `%path%` to insert a new directory at the beginning of the path:

```
set path=c:\utils;%path%
```

### Additional information

You can use the APPEND command to influence the search path.

Refer to Chapter 2 for more information on PATH.

*See also* APPEND and SET.

## PAUSE

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☒ | ■ | ☐ |

### Description

This command temporarily halts batch file execution and waits for the user to press a key.

### Usage

```
pause
```

### Options and switches

None.

## Remarks

When DOS encounters a PAUSE command, it suspends the batch file and displays a message telling the user to press any key to continue. When a key is pressed, batch file execution resumes at the line following the PAUSE command.

You can terminate the execution of the batch file execution pressing Ctrl-Break.

Use the CHOICE command to pause batch file execution and not display the standard PAUSE message.

## Example

The following batch file uses ECHO to display a message and then pauses while the user reads the message:

```
@echo off
cls
echo Preparing to print NOTES.TXT.
echo Make sure the printer is connected and online, and
pause
print notes.txt
```

## Additional information

*See also* CHOICE and ECHO.

## Bob's Shareware Collection disks

BATUTIL, from CtrlAlt Associates, provides enhanced messaging and pausing utilities. TurboBat, a batch file compiler in the ExtraDOS Toolbox from Foley Hi-Tech Systems, provides an excellent way to prevent users from tampering with batch files and improves batch file performance.

# POWER

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 7 | ■ | ■ | ■ | ☐ |

## Description

This command reduces power consumption to prolong battery life on portable computers.

## Usage

```
power [adv[:max or :reg or :min]]
```

or

```
power [std] or [off]
```

## Options and switches

When you execute POWER with no switches, the current power configuration is displayed.

`[adv:max]`

This option activates maximum power conservation but can slow down applications.

`[adv:reg]`

This option turns on power conservation. This command does not save as much power as the MAX switch does, but it has only a limited impact on application performance. This switch is the default.

`[adv:min]`

This option implements modest power savings but has little impact on performance.

`[adv std]`

Use this switch on systems with Advanced Power Management (APM) hardware to instruct POWER to use the hardware-specific power-management facilities.

`[adv off]`

This option turns off the POWER utility.

## Remarks

Before you can use POWER as a command, you must install it as a device driver in CONFIG.SYS (see POWER.EXE). You use POWER from the command line only to change the settings that were set at boot time in CONFIG.SYS.

If you use the STD switch on a computer that doesn't have APM hardware, power conservation is turned off.

### Examples

To see the current power settings, enter the following command:

```
power
```

To maximize power savings, enter the following command:

```
power adv:max
```

### Additional information

CROSS REFERENCE

*See also* POWER.EXE.

## POWER.EXE

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

DOS 6

This device driver conserves power on battery-powered computers.

### Usage

device[high]=[*drive:*][*path*]power.exe [adv[:max or :reg or :min] or [adv std] or [adv off] [/low]

### Options and switches

[*drive:*][*path*]

This option identifies the directory in which POWER.EXE is located, if it is not in the boot directory. SETUP places POWER.EXE in the \DOS directory by default.

[adv:max]

This option activates maximum power conservation but can slow down applications.

```
[adv:reg]
```

This option turns on power conservation. This command does not save as much power as the MAX switch, but it has only a limited impact on application performance. This switch is the default.

```
[adv:min]
```

This option implements modest power savings but has little impact on performance.

```
[adv std]
```

Use this switch on systems with Advanced Power Management (APM) hardware to instruct POWER to use the hardware-specific power-management facilities.

```
[adv off]
```

This option turns off the POWER utility.

```
[/low]
```

This switch forces the device driver to be loaded in base memory.

## Remarks

By default, POWER automatically loads into upper memory, so it should be installed using the DEVICE= statement rather than DEVICEHIGH=.

If you use the STD switch on a computer that doesn't have APM hardware, power conservation is turned off.

You can subsequently change power settings by executing POWER from the command line.

## Examples

To load POWER by using the default settings, add the following line to CONFIG.SYS:

```
device=c:\dos\power.exe
```

To set power consumption to the maximum but install the device drive in base memory, add the following line to CONFIG.SYS:

```
device=c:\dos\power.exe adv:max /low
```

CROSS REFERENCE

## Additional information

*See also* DEVICE and POWER.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **PRINT** | 5 | ■ | ☒ | ☒ | ☐ |

## Description

This command provides background printing capabilities that enable you to proceed with tasks while one or more files is being printed.

## Usage

```
print [/d:device] [/b:size] [/u:ticks1] [/m:ticks2] [/s:ticks3]
[/q:numfiles] [/t] [[drive:][path][filename.ext[ ...]] [/c] [/p]
```

## Options and switches

When you use PRINT with no switches, the status of the print queue is displayed. If print has not been executed in this session, it is installed in memory by using default settings.

`[/d:device]`

This switch specifies the device to use for printing (LPT1, LPT2, LPT3, PRN, COM1, COM2, COM3, or COM4).

`[/b:size]`

This switch sets the size of the print buffer in bytes. Any value in the range from 512 to 16384 is accepted, and the default is 512. Increasing the value speeds up print times.

`[/u:ticks1]`

This switch sets the number of clock ticks PRINT waits to get a response from the print device when it begins to print a job. Any value in the range from 1 to 255 is accepted, and the default is 1.

`[/m:ticks2]`

This switch sets the number of clock ticks PRINT tries to print a character. Any value in the range from 1 to 255 is accepted, and the default is 2.

`[/s:`*ticks3*`]`

This switch sets the number of clock ticks the CPU allocates for the print task. Any value in the range from 1 to 255 is accepted, and the default is 8. Increasing the value improves print times but slows down the foreground task.

`[/q:`*numfiles*`]`

This switch specifies the maximum number of files that can be queued while they are waiting to be printed. Any value in the range from 4 to 32 is accepted, and the default is 10.

`[/t]`

This switch removes all files from the print queue.

`[`*drive:*`][`*path*`]`*filename.ext* `[ ...]`

In a list of one or more files to be printed, each filename must be separated by a space, and wildcards are not supported.

`[/c]`

This switch indicates that the filename preceding the switch and all filenames following the switch are removed from the print queue.

`[/p]`

The /P switch indicates that the filename preceding the switch and all filenames following the switch are added to the print queue. This switch is designed for use at the same time as the /C switch.

## Remarks

You should not use PRINT, designed to print only ASCII files, to print binary files such as word processing documents or spreadsheets. PRINT does not support PostScript printers.

There are approximately 18.2 clock ticks per second.

PRINT automatically expands tabs with eight spaces and finishes printing a file when it encounters the end-of-file character (value 26).

You can use the switches /D, /B, /U, /M, /S, and /Q only the first time PRINT is executed in a session.

Do not delete or rename your file while it is printing or in the queue you'll confuse PRINT.

## Examples

To print the configuration files, enter the following command:

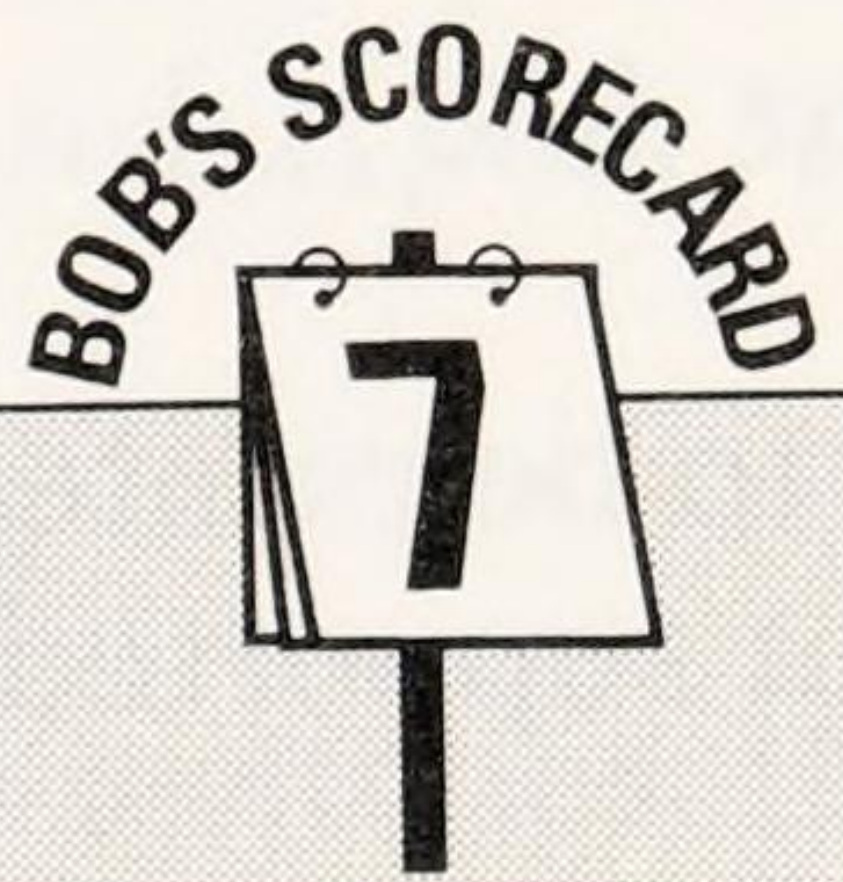
```
print c:\config.sys c:\autoexec.bat
```

To cancel all pending print jobs, enter the following command:

```
print /t
```

## Additional information

The NET USE command can provide access to another computer's printer.



| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **PROMPT** | ☒ | ■ | ☒ | ☐ |

## Description

This command customizes the prompt DOS displays when it is ready to receive a command.

## Usage

`prompt [`*text*`]`

## Options and switches

[*text*]

This option specifies the text to be displayed as the command prompt. The text can include special character codes to instruct DOS to substitute some dynamic text or a line feed.

## Remarks

Every computer with a hard disk should have a PROMPT statement in AUTOEXEC.BAT.

The following table shows the special character codes supported by PROMPT.

| *Code* | *Example* | *Description* |
|---|---|---|
| $b | \| | The \| character |
| $d | Sun 03-21-1993 | The current day and date |
| $e | | The Esc character (used when ANSI.SYS loaded) |
| $g | > | The > character |
| $h | | Erases the preceding character; like Backspace |
| $l | < | The < character |
| $n | D | The default drive |
| $p | C:\DOS | The default drive and directory |
| $q | = | The = character |
| $t | 10:58:15.73 | The time in hours:minutes:second:hundredths |
| $v | MS-DOS Version 6.00 | The DOS version number |
| $$ | $ | The $ character |
| $_ | | Moves to the beginning of the next line |

The prompt is stored in the DOS environment by using the name PROMPT.

If you change to an empty floppy drive and have the path ($p) set in the PROMPT, DOS generates the following message:

```
Abort, Retry, Fail?
```

because it cannot ascertain the default directory on the floppy disk drive. Just enter F, for fail.

## Examples

You establish the most popular prompt, which shows the default drive followed by the > character, by executing the following command:

```
prompt $p$g
```

Use the following prompt statement to generate a two-line prompt that shows the DOS version number on the first line and the text `Enter a command ===>` on the second line:

```
prompt $v$_Enter a command ===$g
```

Use the following prompt with the Backspace facility to prune off the year from a date prompt:

```
prompt $d$h$h$h$h$h$p
```

## Additional information

CROSS REFERENCE

You can enhance PROMPT by using the ANSI.SYS device driver. Refer to Chapter 6 for more details.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **QBASIC** | 7 | ■ | ☒ | ☒ | ☐ |

## Description

This command invokes the full-screen development environment for creating and running basic BASIC language programs.

## Usage

```
qbasic [/b] [/g] [/h] [/mbf] [/nohi] [[/run]
[drive:][path]filename.ext]
```

## Options and switches

```
[/b]
```

This switch suppresses colors and uses black and white.

```
[/g]
```

Use this switch on CGA systems to improve screen-display speeds.

```
[/h]
```

This switch sets the display to 43- or 50-line mode on EGA and VGA systems, respectively.

```
[/mbf]
```

This switch instructs QBASIC to use the function names MKDMBF$, CVSMBF, and CVDMBF rather than MKD$, CVS, and CVD.

```
[/nohi]
```

This switch forces the use of the low-intensity colors.

`[/run]`

This switch loads and executes the specified program.

[*drive:*][*path*]*filename.ext*

This option specifies the name of the program source file to load. If an extension is not specified, QBASIC assumes that the file has a BAS extension.

## Remarks

In addition to providing the QBASIC interpreter, the QBASIC.EXE program provides two important DOS services. The program EDIT.COM calls QBASIC by using the /EDITOR switch, which instructs QBASIC to behave as an editor with no programming facilities. Also, QBASIC is called on by HELP to provide the full-screen on-line DOS help. Behind the scenes, HELP.COM calls QBASIC with the /QHELP switch (must be uppercase).

Don't delete the QBASIC.EXE file, even if you never intend to program in BASIC. Without QBASIC, EDIT and HELP are useless.

## Examples

To start QBASIC, enter the following command:

```
qbasic
```

To run the program GORILLA.BAS, enter the following command:

```
qbasic /run gorilla
```

# RAMDRIVE.SYS

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

## Description

This command creates a RAM disk (a virtual disk in memory) to improve performance.

## Usage

```
device[high]=[drive:][path]ramdrive.sys [disksize sectorsize
[numentries]] [/e] [/a]
```

## Options and switches

[*drive:*][*path*]

This option identifies the directory in which RAMDRIVE.SYS is located, if it is not in the boot directory. SETUP places RAMDRIVE.SYS in the \DOS directory by default.

[*disksize*]

This option specifies the size of the RAM disk in kilobytes. Any value in the range from 4 to 32767 is accepted, and the default is 64.

[*sectorsize*]

This option specifies the size, in bytes, of each sector. Three sizes are supported: 128, 256, and the default 512.

[*numentries*]

This option specifies the maximum number of files and directories the RAM drive can support. Any value in the range from 2 to 1024 is accepted, and the default is a modest 64.

`[/e]`

This switch creates the disk in extended memory.

`[/a]`

This switch creates the disk in expanded memory.

## Remarks

RAM disks were most popular when very few applications could take advantage of extended memory. By using extended memory for a RAM disk, you could not only find a use for the expensive memory but also speed up disk-intensive applications.

The popularity of RAM disks has waned with the emergence of high-performance disk caches. They, too, use memory to improve the performance of disk-related tasks, but they work automatically with all files, without the need to copy files to the RAM disk or the need to remember to save changed information to your real hard drive before powering down.

If you do not specify /E or /A, the RAM disk is created by using base memory.

### Example

Use the following line in CONFIG.SYS to create a RAM disk of 1.45 megabytes, using extended memory:

```
device=c:\dos\ramdrive.sys 1450 /e
```

### Additional information

*See also* DEVICE, DEVICEHIGH, EMM386.EXE, and HIMEM.SYS.

## RD (RMDIR)

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

This command removes (that is, deletes) an empty directory.

### Usage

RD [*drive:*]*path*

### Options and switches

[*drive:*]*path*

This option specifies the name of the directory to be removed.

### Remarks

RD deletes only directories that are completely empty (that do not contain any files or directories). Use DELTREE to delete a directory and all the files and directories contained in it.

If the path begins with .. (dot dot), the directory extending from the parent directory is removed; if the path begins with a \ (backslash), the directory is deleted from the root directory; otherwise, the directory extending from the current directory is deleted.

You cannot remove the current directory — you must change first to a different directory. You cannot remove the root directory.

DOS also supports the long name of the command, RMDIR.

## Examples

To remove the OLDSTUFF directory, enter the following command:

```
rd oldstuff
```

To remove the SAMPLES directory from drive A, enter the following command:

```
rd a:\samples
```

## Additional information

You can manage directories more visually by using the DOSSHELL.

*See also* CD, DELTREE, and MD.

## Bob's Shareware Collection disks

CMFILER, from NoVaSoft, includes advanced directory-management tools, including the capability to prune part of a directory tree and graft it to a new parent directory.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **REM** | 7 | ☐ | ■ | ■ | ■ |

## Description

This command provides a way to add comments to any batch file and CONFIG.SYS.

## Usage

```
REM [text]
```

## Options and switches

[*text*]

This option specifies any comment or string, but it must not include the characters >, <, or |.

## Remarks

REM is short for *rem*ark.

Any text or commands on a line that start with REM are ignored.

## Examples

The following batch file starts with some comments that describe the purpose of the file:

```
rem XYZ.BAT by R D Ainsbury
rem Description: turns on the external lights at night!
....
```

In the following CONFIG.SYS example, REM is used to temporarily disable the mouse command:

```
FILES=20
BUFFERS=15
rem DEVICE=C:\WINDOWS\MOUSE.SYS /Y
```

## Additional information

*See also* ECHO.

BOB'S SCORECARD 6

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **RENAME (REN)** | ■ | ☒ | ☒ | ☐ |

## Description

This command changes the name of one or more files.

## Usage

```
rename [drive:][path]wildcards.ext wildcardt.ext
```

or

```
ren [drive:][path]wildcards.ext wildcardt.ext
```

## Options and switches

```
[drive:][path]wildcards.ext
```

This option specifies the name of the source file or files to be renamed. Wildcards are supported.

*wildcardt.ext*

The new name for the file or files. Wildcards are supported, but every new name must be unique (no other file in the same directory can have the same name).

### Remarks

Renamed files are created in the same directory as the source (you cannot use RENAME to move a file — use MOVE).

You cannot use RENAME to rename a directory — use the DOSSHELL.

### Examples

Use the following command to rename the file THISWEEK.TXT to LASTWEEK.TXT:

```
rename thisweek.txt lastweek.txt
```

To rename all ME files to TXT, enter the following command:

```
rename *.me *.txt
```

### Additional information

*See also* COPY, DOSSHELL, LABEL, and MOVE.

### Bob's Shareware Collection disks

The file manager CMFILER, from NoVaSoft, includes file-renaming facilities and a whole lot more.

BOB'S SCORECARD 6

## REPLACE

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

This special form of the COPY command is designed to synchronize the contents of two directories.

### Usage

```
replace [drive1:][path1]wildcard.ext [drive2:][path2] [/a] [/p] [/r]
[/w] [/s] [/u]
```

## Options and switches

[*drive1:*][*path1*]*wildcard.ext*

This option specifies the source directory and the file or files to be copied.

[*drive2:*][*path2*]

This option specifies the target directory.

`[/a]`

This switch instructs REPLACE to add (copy) files from the source that don't exist in the target. You cannot use this switch with /S or /U.

`[/p]`

Use this switch to force a confirmation prompt before overwriting files or creating new files.

`[/r]`

This switch instructs REPLACE to overwrite read-only files.

`[/w]`

When you specify this switch, REPLACE pauses and waits for you to press a key before commencing.

`[/s]`

Use this switch to make REPLACE search all subdirectories of the target path for matching files.

`[/u]`

This popular switch instructs REPLACE to update only the files in the target path that are older than the same-named files in the source directory.

## Remarks

REPLACE sets the ERRORLEVELs covered in the following table.

| *ERRORLEVEL* | *Description* |
|---|---|
| 0 | Successful operation |
| 1 | Invalid DOS version |
| 2 | No matching source files |
| 3 | Either the source or destination path did not exist |
| 5 | The target files cannot be replaced — they might be read-only |
| 8 | Insufficient memory to run REPLACE |
| 11 | Syntax (command format) error |

### Examples

Use the following command to update all the older common files in the directory C:\HOMEWORK with files from drive A:

```
replace a:\*.* c:\homework /u
```

Use the following command to copy all the new font files from drive B to C:\FONTS:

```
replace a:\*.fnt c:\fonts /a
```

CROSS REFERENCE

### Additional information

*See also* ATTRIB, COPY, MOVE, and XCOPY.

## RESTORE

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

RESTORE is included in DOS 6 to enable you to restore files that were stored on floppy disks by using the original DOS BACKUP utility. BACKUP is no longer included with DOS because it has been replaced by the superior utilities MSBACKUP and MWBACKUP.

### Usage

```
restore drive1: drive2:[path[wildcard.ext]] [/s] [/p] [/b:date]
[/a:date] [/e:time] [/l:time] [/m] [/n] [/d]
```

## Options and switches

*drive1:*

This option specifies the drive that identifies where the backup files are located.

*drive2:*[*path*[*wildcard.ext*]]

This option specifies the name of the drive in which the files will be restored. Files can be restored selectively by using wildcards, but the path must be the same as the path from which the files were originally backed up.

`[/s]`

This switch restores all the matching files in the backup subdirectories and the parent directory.

`[/p]`

This switch instructs RESTORE to prompt for verification before replacing read-only files.

`[/b:`*date*`]`

This switch restores files modified on or before the specified date. The format of the DATE is dependent on the country configuration — use the DATE command to see the correct format for your system.

`[/a:`*date*`]`

This switch restores files modified on or after the specified date.

`[/e:`*time*`]`

This switch restores files modified at or earlier than the specified time. The format of *time* is dependent on the country configuration — use the TIME command to see the correct format for your system.

`[/l:`*time*`]`

This switch restores files modified at or later than the specified time.

`[/m]`

This switch restores only files that have been modified since they were backed up.

```
[/n]
```

This switch restores only files that have been deleted, moved, or renamed since the backup.

```
[/d]
```

This switch performs a dry run by displaying all the files it would RESTORE based on the specified flags and switches.

## Remarks

Unlike earlier versions of RESTORE, the DOS 6 version can restore files from backups created with any version of DOS ranging from DOS 2.0 to DOS 5.0.

RESTORE returns the ERRORLEVELs listed in the following table.

| *ERRORLEVEL* | *Description* |
|---|---|
| 0 | Successful operation |
| 1 | No matching files were found on the backup disk |
| 3 | The user aborted the restoration by pressing Ctrl-Break |
| 4 | An unexpected error occurred |

Do not use RESTORE on drives affected by ASSIGN, JOIN, or SUBST.

## Examples

Use the following command to restore the file BUDGET86.WK1 from drive A to C:\123\PLANS:

```
restore a: c:\123\plans\budget86.wk1
```

Use the following command to restore all changed files in the directory D:\GOALS:

```
restore b: d:\goals\*.* /m
```

## Additional information

CROSS REFERENCE

*See also* MSBACKUP and MWBACKUP.

# RMDIR

*See* RD.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **SET** | 6 | ■ | ■ | ■ | ☐ |

## Description

This command enables you to create, modify, delete, and view environment variable settings.

## Usage

`set` [*varname*=[*string*]]

## Options and switches

When you use SET without switches, the entire environment is displayed.

[*varname*=]

This option specifies the name of the environment variable. Spaces are allowed, but do not use the following characters: <, >, |, :, “, (, ), =, or .

[*string*]

This option specifies the new string to assign to the variable. If no string is specified, the variable is removed from the environment. Avoid the same characters shown for the variable name in the preceding paragraph.

## Remarks

Make sure that you do not include a space before the = character; otherwise, the space forms part of the variable name.

If you get a message that environment space has run out, there is insufficient room to add the specified string. Use the SHELL and COMMAND statements to increase the environment size.

Variable names are not case sensitive, but the strings are stored in the same case in which they are entered.

Batch files can substitute environment strings in commands by specifying the variable name inside two percent characters (%TEMP%, for example).

If you execute COMMAND, a secondary copy of the environment is created. When you use EXIT to return to the original command processor, the original environment is restored.

### Examples

To see the entire environment, enter the following command:

```
set
```

Use the following command to set the TEMP environment variable to E:\TRASH:

```
set temp=e:\trash
```

Use the following command to remove the environment variable TEMP:

```
set temp=
```

### Additional information

*See also* COMMAND, DIR, PATH, PROMPT, and SHELL.

## SETUP

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☐ | ☐ |

### Description

This command runs the DOS 6 installation and setup program.

### Usage

```
setup [/m] [/q] [/e] [/f] [/i] [/b] [/u]
```

### Options and switches

```
[/m]
```

This switch instructs SETUP to run a minimal installation. SETUP installs only the main startup files (IO.SYS, MSDOS.SYS, COMMAND.COM, and DBLSPACE.BIN).

```
[/q]
```

Use the /Q switch to install all the DOS files after you have used the minimal INSTALL switch.

`[/e]`

This switch instructs SETUP to install the optional DOS utilities Anti-Virus, Backup, and Undelete.

`[/f]`

This switch installs DOS 6 on floppy disks.

`[/i]`

This switch skips hardware detection when SETUP is loaded.

`[/b]`

This switch instructs SETUP to display text in black and white.

`[/u]`

This switch forces SETUP to proceed with the installation even when it detects a disk partitioning scheme that may be incompatible with DOS 6.

## Remarks

Unfortunately, SETUP tries to force you to install all DOS files on the boot drive. To create a DOS directory on a different drive, you must first run a minimal installation. You can subsequently use the /Q switch and specify any drive as the target for the external DOS files.

During installation, SETUP requires one (or maybe two) floppy disks to create an UNINSTAL disk.

## Example

To install DOS 6, insert the setup disk in drive A and enter the following command:

```
a:setup
```

## Additional information

CROSS REFERENCE

Refer to Appendix A for more information on SETUP.

*See also* UNINSTAL.

## SETVER

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

Use SETVER if you have programs that expect a specific version of DOS to be installed and that won't run with DOS 6. SETVER displays and modifies the version table in which programs and fake version numbers are stored.

### Usage

```
setver [drive:path] [filename.exe n.nn]
```

or

```
setver [drive:path] [filename.ext [/delete [/quiet]]
```

### Options and switches

When you use SETVER with no parameters or switches, the current contents of the version table appears, showing the program name and the fake DOS version that DOS 6 reports to the program.

```
[drive:path]
```

This option specifies the location of the SETVER.EXE program (*not* the location of the program). By default, SETUP places the file in the \DOS directory.

```
[filename.exe]
```

This option specifies the name of the program (including the extension) DOS should "trick" into believing that a different version of DOS is running.

```
[n.nn]
```

This option specifies the fake DOS version number (5.00, for example). The program name and version number are added to the version table.

```
[/delete]
```

This switch instructs SETVER to remove the program from the version table. The switch can be abbreviated as /D.

```
[/quiet]
```

This switch suppresses the deletion status message.

### Remarks

The version table is effective only when SETVER.EXE (discussed in the next section) is loaded as a device driver in CONFIG.SYS.

Changes to the version table are implemented only the next time the system is booted.

### Examples

Use the following command to display the version table entries:

```
setver | more
```

Use the following command to fool the DUMBCHCK.EXE program into thinking that it is being executed on a DOS 4.01 system:

```
setver dumbchck.exe 4.01
```

Use the following command to remove the METRO.EXE entry from the version table:

```
setver metro.exe /d
```

### Additional information

CROSS REFERENCE

*See also* SETVER.EXE.

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

## SETVER.EXE

### Description

This command loads the version table into memory so that DOS can fool specific programs into thinking that they are being run on a different version of DOS.

### Usage

```
device[high]=[drive:][path]setver.exe
```

### Options and switches

```
[drive:][path]setver.exe
```

This option identifies the directory in which SETVER.EXE is located, if it is not in the boot directory. By default, SETUP places SETVER.EXE in the \DOS directory.

### Remarks

SETVER.EXE is both a device driver and an executable DOS command.

You can modify entries in the version table by running SETVER.

### Example

Use the following statement in CONFIG.SYS to load SETVER.EXE as a device driver:

```
device=c:\dos\setver.exe
```

### Additional information

*See also* SETVER.

## SHARE

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ☒ | ☐ |

### Description

This command allows file sharing and locking for multi-user or multi-application access.

### Usage

```
share [/f:space] [/l:locks]
```

### Options and switches

```
[/f:space]
```

This switch specifies the amount of memory, in bytes, to be reserved for temporarily storing the file-locking details. By default, 2048 bytes are allocated.

`[/l:locks]`

This switch specifies the maximum number of files that can be simultaneously locked. By default, 20 files can be locked at any time.

### Remarks

Because the defaults usually are appropriate, SHARE is typically loaded with no switches.

Use SHARE if you use a network or multitasking software, such as the DOSSHELL, Windows, or DesqView.

You can load SHARE from CONFIG.SYS by using the INSTALL command. Alternatively, on 386 systems and higher, you can conserve base memory by using LOADHIGH to load SHARE into UMBs.

### Example

Use the following command to load SHARE into UMBs:

```
loadhigh c:\dos\share
```

### Additional information

*See also* INSTALL, LOADHIGH, and MEMMAKER.

## SHELL

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 7 | ☐ | ☐ | ☐ | ■ |

### Description

This command sets the options for COMMAND.COM or enables a different command processor to be loaded.

### Usage

`shell=[[drive:]path]filename.ext [parameters]`

### Options and switches

`[[drive:]path]filename.ext`

This option specifies the location and name of the command interpreter (COMMAND.COM, for example).

[*parameters*]

This option specifies optional parameters to specify the characteristics and settings of the command interpreter.

### Remarks

SHELL is used most often in conjunction with COMMAND.COM to change one or more command processor settings (the environment size, for example).

A number of third-party command interpreters, such as 4DOS and NDOS, provide more features than COMMAND.COM. You usually load these interpreters by using the SHELL statement.

MEMMAKER is suspicious of shells other than COMMAND.COM and will disable them by remarking out the SHELL statement containing them. If you're using a third-party command interpreter and run MEMMAKER, you will need to manually edit your CONFIG.SYS file to restore it.

### Example

Use the following statement in CONFIG.SYS to set a larger environment size for COMMAND.COM:

```
shell=c:\dos\command.com c:\dos /e:800
```

### Additional information

*See also* COMMAND.

## SHIFT

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ■ | ☐ |

### Description

This command allows access from a batch file to more than nine passed parameters by dropping the first passed parameter (%1) and shifting all other parameters left one position.

### Usage

```
shift
```

### Options and switches

None.

### Remarks

Before the first SHIFT operation, the %0 parameter identifies the name of the batch file. After the first SHIFT operation, the tenth passed parameter is moved to the %9 variable, the ninth passed parameter is moved to the %8 variable, and so on, until the first passed parameter is moved to the %0 position and the previous %0 value is lost.

You use SHIFT also in batch files in which a single command has to be executed against each passed parameter — SHIFT provides a convenient looping tool (see the following COPYBAK example).

### Example

The following batch file, COPYBAK, uses SHIFT to process multiple filenames passed to the batch file and to copy the specified files to the C:\BAK directory:

```
@echo off
:start
if "%1"=="" goto finish
copy %1 c:\bak
shift
goto start
:finish
```

### Additional information

Refer to Chapter 10 for a discussion of advanced batch file techniques.

### Bob's Shareware Collection disks

TurboBat, in the Extra DOS Toolbox from Foley Hi-Tech Systems, can compile batch files to make them secure and improve execution speed.

## SIZER

MEMMAKER uses this command during the optimization process; it is not intended for general use.

# SMARTDRV.EXE

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 8 | ■ | ■ | ■ | ☐ |

## Description

This command installs or configures the extended memory disk cache.

## Usage

```
smartdrv [[drive[+ or -] ...] [/e:elementsize] [initcachesize]
[wincachesize] [/b:buffersize] [/c] [/r] [/l] [/q] [/s]
```

## Options and switches

```
[drive[+ or -] ...
```

This option specifies the letter of one or more drives. Every drive specified with a plus sign (+) is read cached and write cached; every drive specified with a minus sign (–) has caching disabled; if neither plus nor minus is specified, the drive has read caching but not write caching. If no drive is specified, floppy disks are read cached, and hard drives are read cached and write cached.

```
[/e:elementsize]
```

This switch specifies the amount of cache memory, in bytes, that is managed in each block transfer. The following values are supported: 1024, 2048, 4096, and the default 8192.

```
[initcachesize]
```

This option sets the size of the cache, in kilobytes, that is used when Windows is not running.

```
[wincachesize]
```

This option sets the amount of memory, in kilobytes, that SMARTDRV.EXE relinquishes when Windows is running.

```
[/b:buffersize]
```

This switch sets the size of the read-ahead buffer in kilobytes. The *buffersize* must be an exact multiple of the *elementsize,* and defaults to 16K (16,384 bytes).

`[/c]`

This switch forces cache data to be written immediately to disk. By default, SMARTDRV.EXE uses delayed writes to improve performance, but it can result in data loss if the power shuts off unexpectedly.

`[/r]`

This switch clears the data from the disk cache and restarts SMARTDRV.EXE.

`[/l]`

This switch forces SMARTDRV.EXE to install in base (low) memory. By default, SMARTDRV.EXE tries to load into UMBs when they are available.

`[/q]`

This switch suppresses installation messages when SMARTDRV.EXE is loaded.

`[/s]`

This switch displays a status report.

## Remarks

The following table shows the default values for *initcachesize* and *wincachesize* are based on the total amount of installed extended memory, as follows:

| *Extended memory* | *initcachesize* | *wincachesize* |
|---|---|---|
| 1MB (or less) | All extended memory | 0K |
| 2MB (or less) | 1MB | 256K |
| 4MB (or less) | 1MB | 512K |
| Less than 6MB | 2MB | 1MB |
| More than 6MB | 2MB | 2MB |

SMARTDRV.EXE does not support network or CD-ROM drives, but INTERLNK drives can be read cached.

After SMARTDRV.EXE has been installed, you can execute it additional times, but only the DRIVE+/–, /C, and /R switches are valid.

SMARTDRV.EXE uses extended memory for the cache, but only when HIMEM.SYS (or a similar utility) is installed in CONFIG.SYS.

If you have DoubleSpace volumes, use SMARTDRV.EXE to cache the physical drive (the drive that contains the hidden file DBLSPACE.*nnn*). If you cache both the physical drive and the compressed volume, cache performance is impaired.

## Examples

Use the following statement in AUTOEXEC.BAT with the default values to install SMARTDRV.EXE; SMARTDRV.EXE displays additional status information when it has been loaded:

```
c:\dos\smartdrv /s
```

Use the following statement to install a 4MB cache and specify that 2MB should be relinquished when Windows is running:

```
c:\dos\smartdrv 4096 2048
```

## Additional information

CROSS REFERENCE

Chapter 4 provides more insights into using SMARTDRV.EXE and suggests additional performance-enhancing techniques.

*See also* BUFFERS, EMM386.EXE, FASTOPEN, HIMEM.SYS, and RAMDRIVE.SYS.

## Extended use

In addition to being an executable program, SMARTDRV.EXE can be loaded as a device driver in CONFIG.SYS. The only purpose for using SMARTDRV.EXE as a device driver is to install double buffering by using the following syntax:

```
device=c:\dos\smartdrv.exe /double_buffer
```

Double buffering involves moving data first from the disk cache to a conventional memory buffer and then off to disk. This technique is needed on systems that use a bus-mastering disk controller, such as the ones used for SCSI devices.

If you do not know whether your system has to use double buffering, you can find out easily. Install SMARTDRV.EXE in CONFIG.SYS with the /DOUBLE_BUFFER switch and also in AUTOEXEC.BAT. Use your system for a while, and then execute SMARTDRV with no switches to see a summary of the cache statistics. If every drive includes a `no` in the buffering column, double buffering is not necessary, and you can remove the SMARTDRV.EXE statement from CONFIG.SYS.

CAUTION

By default, SMARTDRV.EXE uses write caching on hard drives. In this mode, SMARTDRV.EXE does not immediately write the data to disk but does so within a few seconds when disk activity slows down. This action provides tangible performance improvement but increases the potential for data loss. If your system power is switched off (or if the reset button is pressed, a program hangs, or the system loses power), the data may not have been written to disk. Do not switch your system off immediately after performing an operation that involves saving data.

You can disable the write cache by using the *drive-* switch.

## SORT

BOB'S SCORECARD 5

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **SORT** | ■ | ☐ | ☒ | ☐ |

### Description

This command sorts input data and directs the sorted output to a device or file.

### Usage

sort [/r] [/+*n*] [<] [*drive1:*][*path1*]*filename1.ext*
[> [*drive2:*][*path2*]*filename2.ext*]

or

[*command* |] sort [/r] [/+*n*] [> [*drive2:*][*path2*]*filename2.ext*]

### Options and switches

[/r]

This switch forces a descending-order sort (from 9 to 0 and from Z to A).

[/+*n*]

This switch instructs SORT to base the sort on a specific column (the character located *n* characters from the left margin, for example).

[*drive1:*][*path1*]*filename1.ext*

This option specifies the file containing the base text that has to be sorted.

*[drive2:][path2]filename2.ext*

This option specifies the new file in which the sorted data is saved.

The > and < characters are pipes that pass the output from one element as the input to the other. You use the | filter to pass the output from a command as input to SORT.

## Remarks

SORT is not case sensitive and should be used on only ASCII files and data.

SORT was often used to sort directory listings, but the DIR /O switch has made that old DOS Jedi trick redundant.

SORT can handle input of only 64K or less.

## Examples

Use the following command to sort the records in the ASCII file NAMES.TXT and place the sorted list in the file ADDRESS.TXT:

```
sort < names.txt > address.txt
```

If the last name begins in column ten, you can sort the data by last name with the following command:

```
sort /+10 < names.txt > address.txt
```

Use the following command to display the sorted list on-screen:

```
sort < names.txt
```

## Additional information

CROSS REFERENCE

*See also* DIR, MORE.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **STACKS** | 5 | ☐ | ☐ | ☐ | ■ |

## Description

This command allocates memory to manage hardware interrupts.

## Usage

```
stacks=n,s
```

## Options and switches

*n*

This option specifies the number of stack frames. Accepted values are 0 or any number in the range from 8 to 64.

*s*

This option specifies the size, in bytes, of each frame. Accepted values are 0 or any number in the range from 32 through 512.

## Remarks

On XTs, original PCs, and IBM PC-portables, the default STACKS setting is 0,0; on all later models, the default is 9,128.

If you encounter stack error messages, such as `Internal Stack Overflow`, you may need to add a STACKS statement to CONFIG.SYS or increase the number and size of the stack frames.

To conserve memory, you may want to experiment with a STACKS setting of 0,0. This setting can save more than 2,000 bytes of base memory.

## Examples

Use the following statement in CONFIG.SYS to eliminate any memory allocation for STACKS:

```
stacks=0,0
```

Use the following statement to implement nine stack frames, each of which is 256 bytes:

```
stacks=9,256
```

## Additional information

CROSS REFERENCE

Refer to Chapter 5 for other memory-saving techniques.

# SUBMENU

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

DOS 6

## Description

This command provides a way to branch to a secondary menu in a multiconfiguration system.

## Usage

`submenu=`*blockname*`[,`*itemtext*`]`

## Options and switches

*blockname*

This option specifies the name of another menu block in CONFIG.SYS that defines a secondary menu. Elsewhere in the file, a unique block heading must have the format [*blockname*]. *blockname* is not case sensitive.

[, *itemtext*]

The blockname is displayed by default as the item description in the menu. Optionally, you can specify a different item description by using the *itemtext* parameter. This description can include spaces and is displayed in the same case in which it is entered. The text can contain as many as 70 characters.

## Remarks

Do not assign a blockname of *menu* or *common* because these blocknames are reserved.

Every item in the menu topic is automatically prefixed with a number.

## Example

The following extract from CONFIG.SYS defines a two-menu configuration file:

```
[menu]
menucolor=14,1
menuitem=Plain,Totally Plain
menuitem=Gainers,Big Gainers
submenu=Support,Load Peripheral Drivers
menudefault=Gainers,9
```

```
[support]
menuitem=Tablet, Tablet Only
menuitem=Cdrom, CD-ROM Only
menuitem=Tplusc, Tablet and CD-ROM
....
```

### Additional information

Refer to Chapter 8 for a thorough description of the multiconfiguration commands.

*See also* MENUCOLOR, MENUDEFAULT, and MENUITEM.

## SUBST

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

### Description

This command assigns a drive letter alias to a specific directory.

### Usage

subst [*drive1:* [*drive2:*[*path*]]]

or

subst *drive1:* /d

### Options and switches

When you use SUBST with no switches, a list of substituted drives is displayed.

*drive1:*

This option specifies the drive letter that will be an alias to the real path.

*drive2:*[*path*]

This option specifies the path that will be assigned to the drive letter alias.

/d

This switch removes the drive substitution for the specified drive.

### Remarks

The drive letter assigned to the substituted path must be less than or equal to the LASTDRIVE.

Do not create SUBST drives from within Windows.

To change the path assigned to a SUBST drive, you must first delete the drive with the /D switch and then reassign the drive with a new SUBST command.

### Examples

Use the following command to substitute drive letter D for the path C:\COMPILER\C700\BIN:

```
subst d: c:\compiler\c700\bin
```

Use the following command to remove the substituted drive D:

```
subst d: /d
```

### Additional information

*See also* LASTDRIVE.

## SWITCHES

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☐ | ☐ | ☐ | ■ |

### Description

SWITCHES is a miscellaneous collection of four configuration switches for customizing the behavior of the operating system.

### Usage

```
switches=[/w] [/k] [/n] [/f]
```

### Options and switches

```
[/w]
```

Use this switch if you are running Windows 3.0 and want to move the file WINA20.SYS from the root directory of the boot drive.

`[/k]`

This switch instructs DOS to treat an Enhanced Keyboard as a standard keyboard. Use this switch only if a program cannot run with an Enhanced Keyboard.

DOS 6

`[/n]`

This switch prevents you from using the F5 and F8 keys at start-up. Use this switch to prevent users from bypassing your CONFIG.SYS and AUTOEXEC.BAT commands.

DOS 6

`[/f]`

This switch instructs DOS not to wait two seconds after displaying the message that MS-DOS is starting. You normally use this switch in conjunction with /N.

## Remarks

If you use the /W switch, you must add the following statement in the [386enh] block of SYSTEM.INI to tell Windows 3.0 where the file is located:

If you use the /K switch with SWITCHES and you want to load ANSI.SYS, you must also use the /K switch with ANSI.SYS.

The SWITCHES statement does not have to be located at the beginning of CONFIG.SYS. DOS scans the file for a SWITCHES statement before displaying the message that MS-DOS is starting.

## Example

To disable the F5 and F8 keys and suppress the two-second delay, enter the following line in CONFIG.SYS:

```
switches=/f /n
```

## Additional information

CROSS REFERENCE

Refer to Chapter 8 for more information on SWITCHES.

*See also* ANSI.SYS.

# SYS

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

## Description

This command makes a formatted drive bootable or updates the DOS version by transferring the primary DOS files IO.SYS, MSDOS.SYS, COMMAND.COM, and DBLSPACE.BIN to the root directory of a drive.

## Usage

`sys [`*drive1:*`][`*path*`]` *drive2:*

## Options and switches

[*drive1:*][*path*]

This option specifies the drive from which the new versions of the DOS files are to be copied. If no drive is specified, SYS looks in the root directory of the default drive.

*drive2:*

This option specifies the drive in which the files will be copied.

## Remarks

SYS transfers files to only local drives (not to network drives) and not to drives affected by ASSIGN, JOIN, or SUBST.

The system files also are transferred to a drive when it is formatted with the /S switch.

## Examples

To transfer the primary DOS files from the disk in drive A to drive C, enter the following command:

```
sys a: c:
```

To transfer the system files from the current drive to the disk in drive B, enter the following command:

```
sys b:
```

## Additional information

*See also* FORMAT.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **TIME** | 5 | ■ | ☒ | ☒ | ☐ |

## Description

This command displays and sets the system time.

## Usage

```
time [hours[:minutes[:seconds[.hundredths]]] [a or p]]
```

## Options and switches

When you execute TIME with no parameters, the current time is displayed and you are prompted to enter a new time. Press Ctrl-Break or Esc and then Enter to return to the DOS prompt.

[*hours*]

This option specifies the current hour, in the range from 0 to 23.

[*minutes*]

This option specifies the current minute, in the range from 0 to 59.

[*seconds*]

This option specifies the current second, in the range from 0 to 59.

[*hundredths*]

This option specifies the current hundredths of a second, in the range from 0 through 99.

`[a or p]`

This option indicates whether the time entered is a.m. or p.m. The hour must be less than 13 to use an *a*. By default, DOS assumes a.m. for all values less than 12.

## Remarks

The TIME format is dictated by the active country setting in CONFIG.SYS.

You do not have to specify the complete time. I usually enter only the hour and minute. I am not too concerned if the system clock (or any clock, for that matter) is accurate to the second.

If your computer does not have a battery-supported clock, you have to enter the correct date and time every time you boot your computer. If so, add DATE and TIME commands to AUTOEXEC.BAT.

## Examples

You can use one of the following two statements to set the time to 3:18 p.m.:

```
time 15:18

time 3:18p
```

I always get confused about how to enter the time between 12 noon and 1 p.m. The following command sets the clock to 25 minutes past noon:

```
time 12:25p
```

The following two commands show how the time can be set to 25 minutes past midnight:

```
time 12:25a

time 0:25
```

## Additional information

CROSS REFERENCE

*See also* DATE.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **TREE** | 6 | ■ | ☐ | ■ | ☐ |

## Description

This command displays the directory structure of an entire drive or for a specific directory and its subdirectories.

## Usage

```
tree [drive:][path] [/f] [/a]
```

## Options and switches

`[`*drive:*`][`*path*`]`

This option specifies the drive or path whose directory structure you want displayed. By default, TREE uses the current directory.

`[/f]`

This switch lists all the files located in each directory.

`[/a]`

This switch instructs TREE to use standard ASCII characters rather than line-drawing characters to draw the directory structure diagram. This switch is normally used when you are redirecting output to the printer, and the printer prints foreign characters rather than lines and corners.

## Remarks

Use the | MORE pipe to pause the display after each screen.

## Examples

Use the following statement to produce a directory tree diagram of the directory C:\WP:

```
tree c:\wp
```

To view the entire directory structure of drive C, enter the following command:

```
tree c:\ |more
```

To print the entire directory structure of drive C to a printer that doesn't support line-drawing characters, enter the following command:

```
tree c:\ /a> prn
```

## Additional information

*See also* CD, DELTREE, DIR, and MORE.

## Bob's Shareware Collection disks

You can switch the CMFILER file manager, from NoVaSoft, into Tree Mode by pressing Shift-T. In this mode, you can traverse the entire drive by using the tree and still use all the tagging and file-management facilities. You can even highlight directories that contain files whose names match a specified wildcard.

LCD, from McAfee Associates, provides an easy way to change directories.

|  | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **TYPE** | 5 | ■ | ☒ | ■ | ☐ |

## Description

This command displays the contents of an ASCII file.

## Usage

`type [`*`drive:`*`][`*`path`*`]`*`filename.ext`*

## Options and switches

[*drive:*][*path*]*filename.ext*

This option specifies the name of the file whose contents will be displayed. If you do not specify a path, the file is assumed to be in the current directory.

## Remarks

If the file is too large to display on one screen, use | MORE to pause the display after each screen.

Unfortunately, TYPE is limited in its capabilities: you cannot search for text, you can only scroll forward through the file, and it doesn't handle binary files very well, for example.

You may prefer to use EDIT to browse larger files or preferably use one of the shareware utilities (see the following "Bob's Shareware Collection disks" section).

## Example

To display the contents of CONFIG.SYS, enter the following command:

```
type c:\config.sys | more
```

## Additional information

*See also* EDIT, PRINT, and MORE.

### Bob's Shareware Collection disks

If you want to view files, you have only one choice: LIST, from Buerg Software, is the champion file browser. You can page up *and* page back through a file, search for text, view in hexadecimal, display a ruler, leave, and jump to bookmarks.

CMFILER, the first-class file manager from NoVaSoft, has the capability to view files.

|  | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **UNDELETE** | 8 | ■ | ■ | ■ | □ |

## Description

This command provides mechanisms for enhancing the chances of recovering deleted files and tries to undelete files that have been deleted.

## Usage

To install the UNDELETE memory-resident tools or review the status of deleted files, use the following syntax:

```
undelete /list or /all or /purge [drive:] or /status or /load or
/u or /s[drive] or /t[drive[-entries]
```

To undelete one or more files, use the following syntax:

```
undelete [[drive:][path]wildcard.ext] [/dt or /ds or /dos]
```

## Options and switches

When you use UNDELETE with no switches or parameters, UNDELETE attempts to restore all deleted files in the current directory but prompts for confirmation before trying to undelete each file it finds.

`/list`

This switch displays a list of all the deleted files.

`/all`

This switch tries to recover all deleted files without prompting for confirmation or prompting for the first character of the filename. A number-sign (#) character is used automatically as the first character. If a file with this name already exists, UNDELETE keeps substituting alternative characters until a unique filename is found.

`/purge [drive:]`

This switch deletes the backup files stored in the sentry directory.

`/status`

This switch shows the type of delete protection currently in force.

`/load`

This switch installs the memory-resident delete-protection portion of UNDELETE by using the setting defined in the DELETE.INI file, which is located in the same directory as UNDELETE.EXE.

`/u`

This switch unloads, and therefore disables, the memory-resident delete-protection facility.

`/s[drive]`

This switch installs the memory-resident delete-protection portion of UNDELETE by using the Delete Sentry protection method. If a drive is not specified, the drives defined in DELETE.INI are used; if DELETE.INI does not exist, the current drive is protected.

`/t[drive[-entries]`

This switch installs the memory-resident delete-protection portion of UNDELETE by using the Delete Tracker protection method. You can optionally specify the maximum number of entries in the tracker file PCTRACKR.DEL. The default value is dependent on the hard disk size and ranges from 25 on 360K floppies to 303 on hard drives greater than 32MB.

You use the following switches to undelete files:

`[[drive:][path]wildcard.ext]`

This option specifies the name of the file or files to be undeleted.

```
[/ds]
```

This switch instructs UNDELETE to try to recover only files that have been protected by using the Delete Sentry method.

```
[/dt]
```

This switch instructs UNDELETE to try to recover only files that have been protected using the Delete Tracker method.

```
[/dos]
```

This switch instructs UNDELETE to try to recover files by using only standard DOS techniques. Use this flag to instruct UNDELETE to ignore the Delete Sentry or Delete Tracker records.

## Remarks

DOS provides the following three levels of deletion protection:

*Delete Sentry.* A hidden directory named \SENTRY is used to store copies of deleted files. The TSR portion of UNDELETE monitors all file-deletion activity and substitutes a command to move the file in place of the file deletion. This form of protection provides the most security because you can undelete the file just by moving it from the deleted directory back to the original directory. On the down side, you don't recover any disk space when you delete a file because it isn't really deleted.

*Delete Tracker.* A hidden file named PCTRACKR.DEL is created in the root directory of the monitored drive. This file contains the name of the deleted file and a list of all the clusters in which the file's data was stored. This method doesn't offer as much protection as the sentry method because other files can still overwrite the clusters. It consumes much less disk space, however, and the chances of recovery are good if there has been limited activity since the file was deleted.

*DOS.* No records are maintained about deleted files, and UNDELETE tries to recover a file's data by using the FAT. When this technique is used, you must identify the filename's first letter because it is modified when the file is deleted. The chances of recovery are greatly reduced by using this method. The odds are improved, however, if the file has recently been deleted, if the file is small, or if the disk is regularly defragmented (so that the file is stored in contiguous clusters). This method does not require that you load UNDELETE as a TSR.

You cannot recover deleted directories with UNDELETE.

Always keep regular file backups as insurance against file loss.

When you install UNDELETE as a TSR (using the /S, /T, or /LOAD switches), you can load it in upper memory with LOADHIGH.

## Examples

You should add the following command to AUTOEXEC.BAT to install Sentry Delete protection for drive C:

```
undelete /sc
```

Use the following command with the sentry method to undelete the file RESIGN.DOC:

```
undelete resign.doc /ds
```

Use the following command to try to undelete all files in the current directory and be prompted to confirm each file:

```
undelete
```

Use the following command to clear (delete, really) all the copies of deleted files stored on drive D:

```
undelete /purge d:
```

## Additional information

Windows users may prefer to use MWUNDEL for general undeletion tasks. Refer to Chapter 3 for a detailed description of UNDELETE.INI and other data-protection and -recovery techniques.

*See also* DEFRAG, DEL, DELTREE, MWUNDEL, and UNFORMAT.

## Bob's Shareware Collection disks

CMFILER, from NoVaSoft, implements a facility similar to Delete Sentry whereby copies of deleted files are retained in a special directory, if the files are deleted by using CMFILER.

To delete a confidential or private file and make sure that nobody UNDELETEs it, use RELDEL, from Tardis DP Consultants. But remember, after it's gone, it's gone!

BOB'S SCORECARD 6

# UNINSTAL

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☐ | ☐ |

## Description

This command removes DOS 6 and restores the previous version of the operating system.

## Usage

```
uninstal
```

## Options and switches

**CAUTION**

UNINSTAL does not support the /? switch. When you enter the UNINSTAL command with the /? switch, the switch will be ignored and UNINSTAL will proceed with the removal of DOS 6. (You can abort the program by pressing F3 twice.)

## Remarks

The UNINSTAL program is copied to the UNINSTAL disk by SETUP when DOS 6 is installed.

To restore the previous version of the operating system, insert the UNINSTAL disk in the boot drive and press Ctrl-Alt-Del.

UNINSTAL uses the files stored in the highest numbered OLD_DOS directory along with the files on the UNINSTAL disk to restore the previous version of the operating system.

You cannot use UNINSTAL to restore the previous version of the operating system if you have done any of the following:

- Run the DELOLDOS command
- Deleted any files in the OLD_DOS directory
- Installed DoubleSpace or any other disk compression program
- Repartitioned the hard drive
- Reformatted the boot drive

**GEM**

You can temporarily boot your system using the previous version of the operating system by inserting the UNINSTAL disk in the boot drive and rebooting. The system will boot from the floppy (using the previous version of the operating system) and automatically start the UNINSTAL program. When UNINSTAL prompts for confirmation, press F3 twice to terminate the UNINSTAL operation. You will be left at the A prompt, having booted with the previous version of the operating system.

CROSS REFERENCE

### Additional information

*See also* SETUP.

BOB'S SCORECARD 8

| | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| **UNFORMAT** | ■ | ☐ | ☒ | ☐ |

### Description

This command recovers files and directories destroyed by an accidental FORMAT.

### Usage

```
unformat drive: [/l] [/test] [/p]
```

### Options and switches

*drive:*

This option specifies the drive to be unformatted.

`[/l]`

This switch instructs UNFORMAT to list every file and subdirectory as they are processed. By default, UNFORMAT displays only fragmented files and subdirectories. Normally, the list scrolls off the screen at a rate of knots. Press Ctrl-S to suspend scrolling, and then press any key to resume.

`[/test]`

This switch performs a dry run that indicates how the disk would be unformatted, but no unformatting is performed.

`[/p]`

This switch directs UNFORMAT output to LPT1.

## Remarks

UNFORMAT cannot unformat a formatted DoubleSpace volume.

UNFORMAT cannot reverse a FORMAT that was performed by using the /U switch. When FORMAT is used *without* the /U switch, a mirror file is created that stores critical UNFORMAT information.

UNFORMAT cannot recover files that are fragmented. You should keep your disk unfragmented by running DEFRAG regularly.

If you have used your disk after using FORMAT, you should consider enhancing your chances for data recovery by using a third-party data-recovery tool such as the Norton Utilities, PC Tools, or the Mace Utilities.

Always keep regular backups of your data for added protection.

## Examples

Use the following command to try to unformat a disk in drive B:

```
unformat b:
```

Use the following command to evaluate the unformat situation for drive D:

```
unformat d: /test
```

## Additional information

Chapter 3 provides more information about data protection and recovery.

*See also* FORMAT, MWUNDEL, and UNDELETE.

# VER

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

## Description

This command displays the DOS version number.

## Usage

```
ver
```

## Options and switches

None.

## Remarks

The MSD diagnostic utility displays the version number.

## Example

To see the DOS version number, enter the following command:

```
ver
```

## Additional information

CROSS REFERENCE

*See also* MSD.

# VERIFY

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ■ |

## Description

This command instructs DOS to verify that data just written to disk can be read back into memory.

## Usage

VERIFY is one of those rare commands that has the same syntax whether it is used in CONFIG.SYS or a batch file or is executed from the DOS prompt.

```
verify [on or off]
```

## Options and switches

When you use VERIFY without a switch, the current VERIFY setting is displayed.

```
on or off
```

Specify ON to instruct DOS to verify all written data or OFF to turn off data verification.

## Remarks

The verification process involves checking to see that the CRC value stored on disk in the sector header is the same as the CRC value of the data written to disk. This process is by no means a thorough verification that the data can be reread from the disk.

I usually leave verification off because it slows down disk throughput and provides only limited safeguards.

### Examples

To see the current verification status, enter the following command:

```
verify
```

To turn on data verification, enter the following command:

```
verify on
```

### Additional information

CROSS REFERENCE

*See also* CHKDSK and COPY.

## VOL

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

This command displays a disk's volume label and serial number.

### Usage

vol [*drive:*]

### Options and switches

[*drive:*]

This option specifies the drive letter of the disk. By default, the current disk is used.

### Remarks

A serial number exists for all disks formatted with DOS Version 4.0 or later.

The same data is printed at the beginning of standard DIR lists.

### Example

Enter the following command to see the volume label and serial number of the default drive:

```
vol
```

### Additional information

Use LABEL to change the volume label. The serial number is assigned automatically when the drive is formatted.

*See also* DIR, FORMAT, and LABEL.

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **VSAFE** | 7 | ■ | ■ | ■ | ☐ |

### Description

This command loads the anti-virus TSR to monitor disk activity and protect against virus infection.

### Usage

```
vsafe [/option[+ or -] ...] [/ne] [/nx] [/ax or /cx] [/n] [/d]
[/u]
```

### Options and switches

When you execute VSAFE with no switches, it is installed with default settings.

```
[/option[+ or -] ...]
```

This switch specifies a number in the range from 1 to 8 that identifies the VSAFE option you want (see the "Remarks" section) followed by the plus sign (+) to enable the option or a minus sign (–) to disable it. You can subsequently modify all eight options from the VSAFE pop-up menu.

```
[/ne]
```

This switch instructs VSAFE *not* to load into expanded memory. When expanded memory is used, VSAFE uses 7K of base memory and 64K of expanded memory.

```
[/nx]
```

This switch instructs VSAFE *not* to load into extended memory. When extended memory is used, VSAFE uses 23K of base memory and 23K of extended memory.

`[/ax or /cx]`

By default, the VSAFE menu can be popped up by pressing Alt-V. Use either the /A*X* or /C*X* switch to instruct VSAFE to pop up on a specific Alt- or Ctrl-key combination, respectively. *X* represents the key you want (/CN for Ctrl-N, for example).

`[/n]`

This switch instructs VSAFE to monitor network drives.

`[/d]`

This switch turns off the checksum option.

`[/u]`

This switch uninstalls VSAFE from memory if no other memory-resident program has been installed since VSAFE. You can uninstall VSAFE by pressing Alt-U when the pop-up menu is displayed.

## Remarks

If no memory options are specified, VSAFE tries to use expanded memory; if no expanded memory is available, VSAFE tries to use extended memory; otherwise, it loads into base memory and uses about 44K.

The following table lists the valid option codes in addition to the default setting and a brief description.

| *Option* | *Default* | *Description* |
|---|---|---|
| 1 | On | Warns before allowing a hard disk format |
| 2 | Off | Warns that a program is trying to load as a TSR |
| 3 | Off | Intercepts all hard disk writes and then prompts for approval before proceeding |
| 4 | On | Checks all program files for viruses as they are opened |
| 5 | On | Monitors all disk boot sectors for viruses |
| 6 | On | Intercepts all attempts to write to a hard disk boot sector or partition table |
| 7 | Off | Intercepts all attempts to write to a floppy disk boot sector |
| 8 | On | Intercepts all attempts to change the contents of a program file |

A number of options force VSAFE to pop up a dialog box that asks for approval before another program can make changes to the disk. If you use Windows with VSAFE installed, you must load the DOS 6 program MSAVTSR.EXE — this program enables VSAFE to pop up a Windows-style dialog box to ask for approval. Ideally, you add the statement LOAD=C:\DOS\MSAVTSR.EXE in WIN.INI to load the program automatically every time you start Windows. Always disable option 3 when you use Windows.

VSAFE does not check for specific brands of viruses. It protects the areas usually infected by viruses and provides a generic protection against all viruses.

If VSAFE suspects that a virus is active and it cannot inform you, it beeps (or, more accurately, tweeps) the PC speaker and reboots the system. If this process occurs, do not log on to a network — run MSAV immediately.

Do not start up VSAFE from Windows.

### Examples

To install VSAFE by using the default settings, enter the following command:

```
vsafe
```

To install VSAFE and disable options 4 and 8 but enable option 7, enter the following command:

```
vsafe /4- /8- /7+
```

### Additional information

Use the program MSAV or MWAV to scan memory and files for known viruses and to remove them.

CROSS REFERENCE

Refer to Chapter 3 for more information on VSAFE.

*See also* BACKUP, MSAV, MWAV, and MWAVTSR.

## XCOPY

BOB'S SCORECARD 8

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ■ | ☐ |

### Description

An enhanced version of COPY that can copy files and directories and that loads multiple files in memory to improve performance.

## Usage

```
xcopy [drive:][path]wildcardS.ext [drive:][path][wildcardT.ext]
[/a or /m] [/d:date] [/p] [/s] [/e] [/v] [/w]
```

## Options and switches

`[drive:][path]wildcardS.ext`

This option identifies the source file or files to be copied. When you specify a path without a filename (c:\dos, for example), the *.* wildcard is assumed. This switch must be specified.

`[drive:][path][wildcardT.ext]`

This option identifies the target location where the files will be copied. If you don't specify a filename, the files retain their original names. If you use a wildcard, the file copies are renamed using the wildcard. If you don't specify a target path or filename, the source files are copied to the current directory.

`[/a]`

This switch copies only matching files that have the archive bit set.

`[/m]`

Like /A, /M copies only matching files that have the archive bit set, but XCOPY also turns the archive bit off after the file is copied.

`[/d:date]`

This switch copies only matching files that have a date stamp equal to or later than the specified date. The format of the date is dependent on the country setting — run the DATE command to see the current format for your system.

`[/p]`

This switch instructs XCOPY to prompt you before creating any directories on the target drive.

`[/s]`

This switch copies all files in all subdirectories from the source path.

```
[/e]
```

Use this switch with /S to instruct XCOPY to create even empty subdirectories on the target path if they exist on the source path.

```
[/v]
```

This switch instructs DOS to verify that the data in the target file can be read successfully. It does not, however, determine whether the target is an accurate copy of the source.

```
[/w]
```

This switch prompts for a key to be pressed before starting the XCOPY.

## Remarks

XCOPY is much faster than COPY because it reads multiple files into memory at one time and then writes them to the target path.

Sometimes an XCOPY command is ambiguous and XCOPY doesn't know whether the target name represents a file or a directory. In these cases, XCOPY asks you to confirm the meaning of the target name.

Unlike COPY, XCOPY does not concatenate multiple files in a single target file — it just copies each file in turn to the target file.

XCOPY returns the following ERRORLEVELs:

| *ERRORLEVEL* | *Message* |
|---|---|
| 0 | Copy completed successfully. |
| 1 | No matching source files. |
| 2 | The user aborted the copy operation by pressing Ctrl-Break. |
| 4 | A file-reading error occurred. |
| 5 | A file-writing error occurred. |

## Examples

Use the following command to copy all files and directories from the disk in drive A and reproduce the directory structure on drive B:

```
xcopy a:\*.* b: /s /e
```

Use the following command to copy all the files in C:\123W\BUDGET to drive I:

```
xcopy c:\123w\budget i:
```

Use the following command to copy all the files that have the file attribute set from the D:\WP\NOTES directory to drive A:

```
xcopy d:\wp\notes a: /m
```

## Additional information

Refer to Chapter 2 for more information on XCOPY.

*See also* ATTRIB, DATE, DISKCOPY, and MOVE.

## Bob's Shareware Collection disks

PCOPY, on of the Patriquin Hard Disk Utilities from Patri-Soft, can do everything XCOPY can, and much more. PCOPY can copy files based on date or size, "best-fit" files to optimally use floppy free space, synchronize source and target disks, prompt before overwriting files, and process all nested subdirectories.

PocketD, the general-purpose utility from Pocketware, provides extensive file copying options.

CMFILER, the file manager from NoVaSoft, makes it easy to tag and copy multiple files.

# Part V:
# Bob's Better-Than-DOS Shareware Collection

6
DOS
Secrets

# Part V: Bob's Better-Than-DOS Shareware Collection

## Introduction

The disks accompanying this book contain some world-class *shareware* programs. Refer to the last page of the book for complete installation instructions.

Nearly all of these programs include text files documenting how to get the most from the software, and these documentation files will be automatically installed to your hard disk along with the executable programs.

Call me old-fashioned, but I much prefer reading printed documentation to browsing text on-screen. This part of the book, therefore, contains information about every shareware program supplied with the book. Unfortunately, there wasn't room to include the complete documentation for every program, so, in most cases, you will find condensed versions of the documentation available on disk.

If you don't find answers to your questions in this part of the book, try looking at the documentation files on disk. Better still, try pressing F1 while the application is running; many of the full-screen programs include comprehensive on-line help.

## Shareware

Shareware programs provide the opportunity for you to receive a fully functional, free version of a program to try out for (usually) a period of about 30 days. At the end of the trial period, you can decide whether you want to keep the program.

If you like the program, register it with the author. With registration, you receive added benefits. To register, fill out an order form (copies are included with each program in this Part) and send your payment to the shareware author. The registration fee for shareware programs depends on the product (usually from $20 to $50) but is usually a nominal sum when you consider the quality of the software.

If you decide not to register the program, remove it from your hard disk and owe nothing. This "honor system" (of paying only if you use the software) is very important if the concept of shareware is going to survive. Shareware authors must be supported if they are going to be able to continue to provide quality software at a minimal cost.

## Benefits of Registration

When you register a shareware program, you usually receive most of the following benefits:

- **A registered version of the software**

  Many registered copies of shareware programs are slightly better than the unregistered versions. For example, with TurboBat (one of the ExtraDOS utilities), the unregistered version has a copyright screen that appears when you first run the program. This screen boldly proclaims that the version being run is an evaluation version only; the registered version replaces this screen with a registered copy designation. Some registered copies of shareware actually contain more features and functionality than their unregistered counterparts.

- **Technical support**

  Most registrations come with telephone, fax, and/or computer bulletin board support.

- **Professionally printed manuals**

  With many shareware programs, you receive a printed manual when you register. This manual, which is more detailed than the on-line printed documentation, often provides illustrations.

- **Free upgrades**

  In most cases, registration gives you the benefit of free upgrades to a future version of the program, with features that may enhance the version you have.

- **Added programs**

  In a few cases, you receive with your registration a disk that contains a registered version of the program, along with other *bonus* shareware programs.

- **License and money-back guarantee**

  Some shareware authors offer you a money-back guarantee with registration. Most registered shareware also offers you a permanent license to use the program on your PC.

- **Source code or custom programming options**

  Some authors give you source code when you register; others offer custom programming options instead. You may have to pay extra for these benefits.

  The biggest overall benefit of registering is that the registration of shareware encourages the development of new shareware programs.

## Where to Get Shareware

Shareware is usually distributed by electronic information services, such as CompuServe, GEnie, and many others, as well as electronic bulletin board services (BBSs). See Appendix E for further information on these electronic services.

## The Association of Shareware Professionals (ASP)

The Association of Shareware Professionals (ASP) was formed in 1987 to "strengthen the future of *shareware* (user-supported software) as an alternative to software distributed under normal retail marketing methods."

Note that the ASP members subscribe to a code of ethics and are committed to the concept of shareware. The ASP certifies programs that meet their criteria for shareware and sponsors events at computer industry events, such as COMDEX and PC EXPO and many others. If you are a software author, the ASP may help you find distribution channels for your program.

The ASP demands professionalism from its members. For example, each ASP author must acknowledge every registration to the purchaser and must provide at least three months of technical support for their products. ASP members must also stand behind any benefits they offer.

Membership dues in the ASP are very reasonable. For more information, write the Executive Director, Association of Shareware Professionals, 545 Grover Road, Muskegon, MI 49442-9427, or send a message to CompuServe 72050,1433.

## The ASP Ombudsman Program

To resolve any questions about the role of shareware, registrations, licenses, and so on, the ASP established an Ombudsman to hear all parties. Not all of the shareware authors who have programs on the *DOS 6 SECRETS* disks (*Bob's Better-Than-DOS Shareware Collection)* are members of the ASP. But if you have a support problem with an author who is an ASP member, and you cannot settle it directly with the author, the Ombudsman may help you find a remedy. Remember that you cannot expect technical support for any program unless you are a registered user of that program.

Assume that the following ASP statement applies to each of the programs on the *DOS 6 SECRETS* disks: "These programs are produced by members of the Association of Shareware Professionals (ASP). ASP wants to make sure that the shareware concept works for you. If you are unable to resolve a shareware-related problem with an ASP member by contacting the member directly, ASP may be able to help. The ASP Ombudsman can help you resolve a dispute or problem with an ASP member, but does not provide technical support for members' products. Please write to the ASP Ombudsman at P.O. Box 5786, Bellevue, WA 98006, or send a CompuServe message to ASP Ombudsman, 70007,3536."

## General Shareware License Agreement

Each of the shareware programs on the *DOS 6SECRETS* disks has its own license agreement and terms. These are printed in the section describing each program, or in a text file enclosed with the program on the disk. Most programs are copyrighted (and assume "All Rights Reserved" even where not explicitly stated). In general, you should assume that any shareware program adheres to at least the following license terms suggested by the ASP, where Program is the specific shareware program, and Company is the program's author or publisher:

> The Program is supplied as is. The author disclaims all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The author assumes no liability for damages, direct or consequential, which may result from the use of the Program.
>
> The Program is a *shareware* program, and is provided at no charge to the user for evaluation. Feel free to share it with your friends, but please do not give it away altered or as part of another system. The essence of *user-supported* software is to provide personal computer users with quality software without high prices, and yet to provide incentives for programmers to continue to develop new products. If you find this program useful, and find that you continue to use the Program after a reasonable trial period, you must make a registration payment to the Company. The registration fee will license one copy for one use on

any one computer at any one time. You must treat this software just like a copyrighted book. An example is that this software may be used by any number of people and may be freely moved from one computer location to another, as long as there is no possibility of it being used at one location while it's being used at another — just as a book cannot be read by two different persons at the same time.

Commercial users of the Program must register and pay for their copies of the Program within 30 days of first use or their license is withdrawn. Site-license arrangements may be made by contacting the Company.

Anyone distributing the Program for any kind of remuneration must first contact the Company at the address provided for authorization. This authorization will be automatically granted to distributors recognized by the ASP as adhering to its guidelines for shareware distributors, and such distributors may begin offering the Program immediately. (However, the Company must still be advised so that the distributor can be kept up-to-date with the latest version of the Program.)

You are encouraged to pass a copy of the Program along to your friends for evaluation. Please encourage them to register their copy, if they find that they can use it. All registered users will receive a copy of the latest version of the Program. You are also encouraged to give feedback to the shareware authors by sending in comments about the existing programs and what you would like to see in the future.

Each of the programs and documentation thereto are published and distributed with this book with the written permission of the authors of each. The programs herein are supplied as is. Robert D. Ainsbury and IDG Books Worldwide Inc. individually and together disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any particular purpose; and assume no liability for damages, direct or consequential, which may result from the use of the programs or reliance on the documentation.

## Software Installation

Complete installation instructions for the DOS SECRETS disks are printed on the last page of this book.

## Disk Contents

The following programs are included on the *DOS 6 SECRETS* disks (*Bob's Better-Than-DOS Shareware Collection).*

## Bob's Better-Than-Shareware Collection Disks: Contents at a Glance

| *Program* | *Description* | *Disk Space Required* |
|---|---|---|
| ANARKEY | Command Line Enhancer | 326K |
| ARJ | File Compression Tools | 395K |
| BATUTIL | Batch File Enhancer | 539K |
| CMFILER | File Manager | 377K |
| DOG | Disk Defragmenter | 119K |
| DUPE | Disk Duplicator | 81K |
| EXTRADOS | | 346K |
| ALARMCLK | Alarm Clock | |
| ALLSUB | Subdirectory Processor | |
| CALENDAR | Calendar Lister | |
| CLEANUP | Hard Disk Cleanser | |
| DISKINFO | Disk Information Utility | |
| NO | File Exclusion Utility | |
| TURBOBAT | Batch File Compiler | |
| TURBOTXT | Text File Compiler | |
| FANSI | ANSI.SYS Replacement | 879K |
| FORMAT MASTER | Disk Formatter | 300K |
| HOTDIR PLUS | Directory Lister | 102K |
| LCD | Directory Changer | 74K |
| LIST | File Browser | 257K |
| PATRI-SOFT | | 414K |
| PCOPY | File Copying Utility | |
| PDAILY | Executes Once-a-Day | |
| PDEL | File Deleting Utility | |
| POCKETD | Directory/Copy/Search+ | 231K |
| QFIND | File Finder | 83K |
| QEDIT | Full-screen Editor | 356K |

| | | |
|---|---|---|
| RELDEL | Permanent File Eraser | 31K |
| SECRETS | Batch Files etc. from Book | 20K |
| SEEKEASY | Text Search Utility | 131K |
| SPINTOOL | Disk Analysis Software | 33K |
| STACKEY | Keyboard Manager | 295K |
| UMBDRV | Memory Manager | 98K |
| VIRUSCAN | McAfee Virus Scanner | 367K |

# ANARKEY

**Version 4.0**
**Copyright © 1988–1991 by Steven Calwas, Moderne Software**

## Introduction

Anarkey is an intelligent command-line editor for DOS. With Anarkey, complete input lines can be entered with two or three keystrokes. Anarkey is intelligent because there is no need to tell it what you want to enter on the command line; simply press the <Tab> key and Anarkey figures out what you want and does it for you.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the Anarkey files in the directory \SECRETS\ANARKEY. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the Anarkey files to a directory that is included in the search path.

The complete Anarkey distribution includes the following files:

| | |
|---|---|
| ANARKEY.COM | Anarkey Program |
| ANARKEY.DOC | Anarkey User Manual |
| ANARKEY.SUM | Summary list of Anarkey key assignments |
| 101-KEYS.COM | Support for Enhanced 101/102-Key Keyboards |
| ANARKED.EXE | The Anarkey Keystroke Editor |
| ANARKMD.EXE | The Anarkey Command-Line Utility |
| CONVERT.EXE | The Anarkey AKA-Conversion Utility |
| LOAD.EXE | The Anarkey Loader |
| CREDIT.CRD | Information on registering by credit card |
| ORDERFRM | Registration form |
| READ.ME | Read this file first |

***Note:*** Only Chapters 1, 2, and 9 are reprinted here. For full documentation, please read the file ANARKEY.DOC.

## Chapter 1: Tutorial

Here is a list of the main operations Anarkey provides:

- Line-edit features comparable to those found in word processors.
- Ability to save input lines for later retrieval.
- Three methods to retrieve saved command lines:
  1. Sequentially walk through the list of saved commands.
  2. Line completion where you enter a portion of the line and Anarkey completes the remainder for you.
  3. Pop-up window which lists all saved commands. Supports a mouse.
- A text substitution capability called an AKA (also referred to as an *alias* or *synonym*). The Convert program is included in the Anarkey package to translate the alias and synonym definitions of other editors to Anarkey's format. AKAs can even be executed from a batch file.
- Enter multiple commands on a single input line.
- A 255-character input line.
- Comprehensive filename completion.
- Environment variable editing.
- Use expanded memory.
- Display a non-blinking cursor.
- Pop-up windows with built-in mouse support to retrieve saved commands and complete filenames as well as to edit environment variables.
- Configurable key assignments.
- A MegaKey which analyzes the input line, determines the type of Anarkey operation you want, and does it for you automatically. The MegaKey is what makes Anarkey an intelligent command-line editor.

- Write the list of saved commands and AKAs to a disk file for future restoration.
- Comprehensive UNIX switchar support.
- Pop-up help window with configurable help text.
- Full support for enhanced 101/102-key keyboards.
- Pop-up windows offer you a choice between two mouse interfaces.
- Special support for the 386 Enhanced Mode of Microsoft Windows 3.0.
- And much more!

Anarkey is written entirely in assembly language for maximum efficiency in memory usage and execution speed. When installed in its default configuration, Anarkey uses about 11K of memory. If expanded memory is available, Anarkey requires only 1K of conventional memory.

## Installation

Anarkey makes significant use of the Anarkmd program. To guarantee that Anarkey works properly at all times, place the ANARKMD.EXE file in a directory specified in the PATH environment variable.

To run Anarkey, execute the ANARKEY.COM program from either the DOS prompt or from your AUTOEXEC.BAT file. Since you will probably want Anarkey active whenever you are using your computer, executing Anarkey from the AUTOEXEC.BAT file makes the most sense.

When run, Anarkey installs itself into the DOS environment and terminates but stays resident. A default configuration is used but can be modified via command-line options. Anarkey options are described in Chapter 11 of the complete online manual. For the remainder of this tutorial, we assume the default configuration is in effect.

Once resident, Anarkey handles all input requests for the following situations:

- The DOS command line
- Redirected input
- DEBUG, SYMDEB, EDLIN, and several third-party programs
- All DOS sessions running under Windows 3.0 in 386 enhanced mode

Notice that Anarkey does not process lines executed from a batch file. However, the Anarkmd program described in Chapter 16 provides an indirect method to support batch files.

***Note:*** Please read the ANARKEY.DOC FILE for the "Quick-Start Tutorial" section omitted here if you need help getting started with Anarkey.

Install Anarkey by executing the ANARKEY.COM program.

Anarkey lets you place the cursor anywhere within the input line and enter or delete characters. Once a line has been saved, it can be retrieved so you won't have to retype it. Pressing the <Up> and <Down> arrow keys "walks" you through the list of saved lines. Once retrieved, a line can be edited using the line-edit functions.

You can display a full-screen window containing the list of the saved commands by pressing <Alt-F4>. (Make sure ANARKMD.EXE is stored somewhere along your PATH variable before doing this.) If you want to enter something on the input line, press the <Tab> key and Anarkey figures out what you want and does it automatically.

## Chapter 2: The Pop-Up Help Window

The help window can be popped up by pressing <Alt-F1> or <Alt-H> whenever Anarkey is active, for example, at the DOS command prompt. Before you pop up the window, make sure the Anarkmd program (in the file ANARKMD.EXE) is in either the current directory or a directory specified in the PATH environment variable. After pressing the help key, a full-screen window will be displayed containing a list of Anarkey functions and their current key assignments.

**Note:** Anarkmd must be in the current directory or along the PATH because it actually performs the entire pop-up window operation. Anarkey simply executes Anarkmd, telling it to display the help window. But all this is performed automatically in the background, and you do not need to worry about it.

The information displayed in the window is actually read from a standard text file just before the window pops up. By reading and displaying a separate text file, Anarkey saves precious DOS memory by not keeping the help information resident. It also provides great flexibility in the content and format of the displayed help information. In other words, you can modify the help text file to your heart's desire, even include help information about other programs if you wish. Only two restrictions apply: (1) the file cannot exceed 65,000 characters in size and (2) the file may not contain more than 1,000 lines of text.

Since a text file must be loaded to gather the help information, Anarkey must be informed of the help file's name. Anarkey first looks for the file name in an environment variable called **HELP@**. If **HELP@** is defined, it should specify the drive, directory, and filename of the helpfile. For example,

```
SET HELP@=c:\anarkey\anarkey.hlp
```

If HELP@ is not defined, Anarkey assumes the help file's name is ANARKEY.SUM and will attempt to locate a file with that name. ANARKEY.SUM is a text file distributed with Anarkey which contains a list of Anarkey functions and their default key assignments. ANARKEY.SUM can also be updated by Anarked, the Anarkey Keystroke Editor, if you modify any of the default key assignments. Anarked is explained in Chapter 13.

Depending upon which version of DOS you are running, Anarkey will search for the ANARKEY.SUM file in two locations. First, if running under DOS version 3.0 or later, Anarkey searches for ANARKEY.SUM in the same directory in which Anarkmd is located, also called the home directory. Under versions of DOS prior to 3.0, Anarkey does not check this location.

If the ANARKEY.SUM file is still not found, Anarkey looks for it in the current directory.

These search steps and the order in which they are performed are outlined below:

1) File specified in HELP@ environment variable
2) ANARKEY.SUM in Anarkmd home directory (DOS 3.0+ only)
3) ANARKEY.SUM in current directory

If all attempts to locate the help file fail, Anarkey displays an error message (`Help file not found`) and ends the help operation. Otherwise, the help file is loaded and displayed in the pop-up window.

With the help window displayed, the following keystrokes are recognized and acted upon:

| | |
|---|---|
| <F1> | Display help for pop-up window |
| <Esc> | Exit the help window |
| <Enter> | Same as <Esc> key |
| <Up> | Scroll the window contents up one line |
| <Down> | Scroll the window contents down one line |
| <PgUp> | Scroll the window contents up one page |
| <PgDn> | Scroll the window contents down one page |
| <Home> | View top of help file |
| <End> | View bottom of help file |

In addition to keystroke input, if a mouse is installed on the computer, it can be used to scroll through the help window. Anarkey supports the Microsoft and any Microsoft-compatible mouse. With the mouse installed, a separate mouse cursor is displayed in the middle of the window in the form of a solid block character. By moving the mouse, the mouse cursor can be moved anywhere on the screen.

Scrolling operations are performed with the mouse by moving the mouse cursor to specific locations in the window and clicking (pressing and releasing) the left mouse button. For example, you can position the mouse cursor anywhere over the

```
Press F1 for help
```

message in the upper right corner of the window and click the left mouse button. This will have the same effect as pressing <F1> in that it will display an additional help screen. Pressing either mouse button will clear this secondary help window.

On the far right edge of the help window is a scroll bar. The scroll bar is actually composed of four separate sections. At the very top of the bar is a single-character up-arrow. To scroll the window contents up one line, position the mouse cursor on this arrow and click the mouse button. A down-arrow is located at the very bottom of the scroll bar. Click on it to scroll the window contents down one line. The remainder of the scroll bar is divided into two equal halves. Click on the top half to scroll the window up one page; click on the bottom half to scroll down one page. You can also press and hold the mouse button with the mouse cursor positioned at any of these locations. Doing so is the same as repeatedly clicking on the item.

***Note:*** To clear the help window, click the right mouse button.

After exiting the help screen, you will be returned to the DOS command prompt. Any characters entered at the prompt when the help window was popped up will still exist.

The mouse movements described in this chapter are also applicable to the other pop-up windows that Anarkey supports. These other windows are discussed in future chapters.

Anarkey also supports an alternative mouse interface (in case you don't like the interface described in this chapter). The alternative mouse interface is described in Appendix I.

***Note:*** See the ANARKEY.DOC file for Chapters 3–8 omitted here.

# Chapter 9: The MegaKey and MenuKey

The MegaKey and MenuKey are two of Anarkey's most potent strengths. With these two simple keys, you gain access to all of Anarkey's completion, line-retrieval, and window functions.

## The MegaKey

In the previous chapters, you learned about several separate Anarkey functions which retrieve and complete items on input lines. Each of those functions and their assigned keystrokes are listed:

| | |
|---|---|
| <F5>, <Ctrl-K> | Retrieve line from history buffer |
| <F7>, <Ctrl-D> | Complete directory name |
| <F8>, <Ctrl-F> | Complete filename |
| <F9>, <Ctrl-E> | Retrieve environment variable |

Rather than force you to remember all those keystrokes and have to think about which one to press at a given time, Anarkey combines them all into a single MegaKey. By default, the MegaKey is assigned to <Tab>. When you press <Tab>, Anarkey determines on its own the type of completion or retrieval operation you desire and does it for you. Of course, you can still press any of the keys listed above, but in most instances, it will be easier to just press <Tab> and let Anarkey do all the work.

Like all the other retrieval and completion functions, the MegaKey can be pressed repeatedly to cycle through the list of possibilities. The MegaKey will correctly determine the desired type of operation an amazingly high percentage of the time. However, you may occasionally have to cycle through some unwanted completions before getting to the one you want.

## The MenuKey

The MenuKey is a close cousin of the MegaKey. Like the MegaKey, it combines several operations to a single key. The MenuKey can be thought of as a front-end to all of Anarkey's pop-up window functions listed below.

| | |
|---|---|
| <Alt-F1>, <Alt-H> | Pop-up help window |
| <Alt-F5>, <Alt-K> | Pop-up line retrieval window |
| <Alt-F8>, <Alt-F> | Pop-up filename completion window |
| <Alt-F9>, <Alt-E> | Pop-up environment window |

The MenuKey is assigned to the <Alt-Tab> and <Shift-Tab> keys. However, be aware that the <Alt-Tab> key is not normally recognized on DOS systems. If your system has an enhanced, 101-key keyboard (i.e., a keyboard with <F11> and <F12> keys), you can use the 101-Keys program to allow DOS to recognize <Alt-Tab> and many other keystrokes. 101-Keys is discussed in Chapter 15.

Like all the other Anarkey window functions, be sure the Anarkmd program is located in either the current directory or a directory along the PATH before pressing the MenuKey.

The MenuKey displays a pop-up menu. Each of the above operations is listed in the menu. In addition, the item *Return to Command Line* is included in the menu. Using the selection techniques discussed for windows in previous chapters, you select a menu item via the keyboard or a mouse. The operation you select is then executed.

For example, if the help window menu item is selected, the pop-up help window is displayed on-screen. The effect is the same as if you had pressed the <Alt-F1> key to display the window, but now you only have to remember a single key, the MenuKey, to display any of the supported Anarkey windows.

## Summary

Despite its brief length, this chapter described Anarkey's most powerful functions, the MegaKey and MenuKey. These two functions are great leaps toward a state of total Anarkey, a state without keystrokes. Learn them and your life at the command prompt will become a pleasure.

| | |
|---|---|
| <Tab> | MegaKey |
| <Alt-Tab> | MenuKey |
| <Shift-Tab> | Same as <Alt-Tab> |

## ANARKEY Function Summary

**Help Window**

| | |
|---|---|
| Help window | <Alt-F1> or <Alt-H> |

**Line Editing Functions**

Cursor movement:

| | |
|---|---|
| To start of line | <Home> |
| To end of line | <End> |
| Right one character | <Right> |
| Right one word | <Ctrl-Right> |
| Left one character | <Left> |
| Left one word | <Ctrl-Left> |

Character deletion:

| | |
|---|---|
| Character left | <Backspace> |
| Current character | <Del> |
| Word right | <Ctrl-T> |
| To end of line | <Ctrl-End> or <Ctrl-Y> |
| Delete line | <Esc> |

Character retrieval:

| | |
|---|---|
| Single char from previous line | <F1> |
| Remainder of previous line | <F3> |
| Remainder of prev line and execute | <Alt-F3> |

Miscellaneous:

| | |
|---|---|
| Accept line | <Enter> |
| Toggle input mode | <Ins> |
| Accept next keystroke unmodified | <Ctrl-U> |

**MegaKey and MenuKey**

| | |
|---|---|
| MegaKey | <Tab> |
| MenuKey | <Alt-Tab> or <Shift-Tab> |

**History Buffer Functions**

| | |
|---|---|
| Retrieve previous buffer line | <Up> |
| Retrieve next buffer line | <Down> |
| Complete line | <F5> or <Ctrl-K> |
| Complete line and accept | <F6> or <Ctrl-L> |
| History-buffer window | <Alt-F5> or <Alt-K> |
| Restore CBL, get next line | <PgDn> |
| Display buffer contents | <F4> |
| Display buffer contents from bottom | <Ctrl-F4> |
| Write buffer to \ANARKEY.LOG | <Ctrl-W> |
| Save line in buffer, do not execute | <Ctrl-J> |
| Remove current buffer line | <Ctrl-Z> |
| Clear buffer | <Ctrl-X> |

**AKA Functions**

| | |
|---|---|
| Display AKAs | <Alt-F4> |
| Undefine all AKAs | <Alt-X> |

**Directory, Program, and File Name Completion Functions**

| | |
|---|---|
| Complete directory name | <F7> or <Ctrl-D> |
| Complete file/program name | <F8> or <Ctrl-F> |

Filename-completion window ........................ <Alt-F8> or <Alt-F>

**Environment Variable Retrieval**

Retrieve environment variable ........................... <F9> or <Ctrl-E>

Environment-variable window ....................... <Alt-F9> or <Alt-E>

**UNIX Switchar Programs**

Display UNIX programs .................. <Shift-F4>

**Miscellaneous Functions**

Check command lengths ........................ <F2>

Repress multiple commands per line ......................... <F10> or <Ctrl-R>

## ANARKEY 4.0 Registration Form

Name: ______________________

Company: ______________________

Address: ______________________

City: ______________________

State: __________ Country: ______________

Zip: __________ Phone: ______________

**Remit to:**
Steven Calwas
Moderne Software
P.O. Box 3638
Santa Clara, CA 95055-3638
(408)247-0509

(3434 Warburton Avenue, Suite 10, 95051)

Please send me a registered Anarkey disk and printed manual.

☐ Anarkey includes disk and printed manual
$29.95 each $ ______

☐ Network License, includes one disk and manual

***Note:*** Although one physical copy of the software is delivered, each network user must obtain a license.

First user ...................................... $29.95 $ ______
Subsequent users .......... x $10.00 each $ ______
CA residents, add sales tax .................. $ ______
Shipping and Handling
U.S.: ............................................. $4.00 $ ______
Outside U.S.: ............................. $8.00 $ ______
Total ........................................................ $ ______

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Payment by:
☐ Check ☐ Money Order
☐ Credit Card (be sure to attach CREDIT.CRD form)

# ARJ

**Version 2.30**
**Copyright © 1990–92 by Robert K. Jung**

## Introduction

ARJ version 2.30 is a full-featured shareware release. The use of ARJ in a business, commercial, government, or institutional environment requires a license. This software as originally released by the author came in an ARJ self-extracting archive with an ARJ-SECURITY envelope to prevent tampering. Re-archival of this software will lose this security feature.

ARJ is the result of my desire to use my interest in compression technology to produce an archiver for personal use on my PC and on minicomputers that provides power and excellent flexibility. I expect ARJ will continually be improved both in speed, compression, and features. There are plans to port versions of ARJ to other platforms in the future pending sufficient time and funding.

***Important:*** Users of ARJ should read the WHATSNEW.DOC and UPDATE.DOC files which contain information about the latest improvements to ARJ.

## Major Features of ARJ

- ARJ is currently the trendsetter in PC file archivers in terms of features and compression.
- ARJ 2.30 currently provides excellent file compression in terms of resulting archive size and speed.
- ARJ includes an option to create multiple volume archives with one command making ARJ useful for making backups to multiple floppies or archiving a very large file to several floppies without SLICEing.
- ARJ includes a myriad of commands and options to provide outstanding flexibility in archiver usage.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the ARJ files in the directory \SECRETS\ARJ. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the ARJ files to a directory that is included in the search path. For instance, to install the ARJ software, you may simply copy ARJ.EXE, REARJ.EXE, REARJ.CFG, and ARJSORT.COM to one of the directories named in your DOS PATH statement found in your AUTOEXEC.BAT.

### File Contents

| | |
|---|---|
| ARJ230.EXE | An ARJ-SECURED self-extracting archive containing the following files: |

| | |
|---|---|
| ARJ.DOC | ARJ user manual |
| ARJ.EXE | Version 2.30 |
| ARJ_BBS.DOC | List of ARJ support BBSes |
| ARJSORT.BAT | Sort archive by various parameters |
| ARJSORT.COM | Compiled ARJSORT.BAT file |
| ARJSORT.DOC | ARJSORT user manual |
| CREDIT.CRD | Form for handling credit card orders |
| LICENSE.DOC | License policy for ARJ |
| ORDERFRM.DOC | Order form for ARJ |
| README.DOC | Read this file first |
| REARJ.CFG | Default REARJ configuration file |
| REARJ.DOC | REARJ user manual |
| REARJ.EXE | Archive conversion utility |
| REGISTER.EXE | Program to register ARJ programs |
| SYSOP.DOC | SYSOP registration special offer |
| TECHNOTE.DOC | ARJ technical information |
| UPDATE.DOC | Update to ARJ user manual |
| WHATSNEW.DOC | Changes to ARJ from 1.00 to now |
| WHY_ARJ.DOC | ARJ commercial |
| ARJBACK.BAT | Batch file to fully back up C: to A: |
| ARJREST.BAT | Batch file to restore to C: from A: |
| ARJUPDAT.BAT | Batch file to incrementally back up C: |
| REARJALL.BAT | Batch file to REARJ ZIP, LZH to ARJ |

## Terminology

The following terms are used throughout this manual.

ARCHIVE — This is a file containing one or more files in a compressed or noncompressed state and containing file-related information such as filename and date-time last modified, etc.

ARJ FILE — This is an archive created by ARJ (sometimes called an *arjive* in slang terminology).

BACKUP TYPE ARCHIVE — This is an ARJ archive which has the internal backup flag turned on. This causes all current and future updates to the archive to *not* overwrite internal files in the archive when adding files with the same names as ones already in the archive. The older duplicate files in the archive will be marked as backups.

COMPRESSION — The process of encoding redundant information into data requiring less storage space.

COMPRESSION PERCENTAGE/RATIO — The percentage compression reported by ARJ is a variation of one of the TWO standard methods of expressing compression ratio in the technical literature. ARJ uses the compressed size/original size ratio. The other method is the inverse ratio. When ARJ reports 96 percent as the compression ratio, that means that the compressed file is 96 percent of the original size (very little compression). Other archivers use their own methods. LHARC uses the same ratio as ARJ.

EXTRACTION or UNCOMPRESSION — The processing of re-creating the exact information that was previously compressed.

SELF-EXTRACTION MODULE (SFX) — This is an archive that is an executable file that is capable of extracting self-contained files.

TEXT MODE — In text mode, ARJ inputs the file using the C library text mode which translates the carriage return, linefeed control characters of MS-DOS to a single linefeed character. This saves space and provides the option for cross platform file extraction. On another platform, the host C library would change the single linefeed to the host text newline separator sequence. In addition, for platforms such as PRIMOS which set bit 8 in ASCII text characters, ARJ sets/resets bit 8 according to the platform extracted to. When extracting a text mode file to the same type of platform archived from, ARJ will *not* strip the 8-bit text to 7-bit text.

VOLUMES — These are ARJ archives that are in sequence and have been created by a single ARJ command. Files in the volumes may span volumes in a split format. These volumes are usable archives.

## Features of ARJ

- Currently ranks as one of the best in compression in terms of size reduction of the currently available archivers including PKZIP 1.10, PAK 2.51, ARC 7.0 (ARC PLUS), LHARC 1.13c, LHA 2.13, and the new ZOO 2.10. ARJ is particularly effective with database files, graphics files, and large documents. With the **-jm or -jm1** option, ARJ usually compresses even smaller at a cost of time.
- Archive and individual file comments with option of inputting comments from a file.
- ARJ has MS-DOS 3.x international language support for the proper casing of filenames and text.
- 32 bit CRC file integrity check.
- DOS volume label support.
- Empty directory support.
- File-generation archive support where ARJ will allow the user to keep several versions of the same file in an archive.
- Test new archive before overwriting the original archive option.
- Multiple volume archives with one ARJ command. This allows the user to back up a full hard disk drive to multiple floppies. Recovery of individual files is convenient because each volume is an individual archive except for split files. No need to use SLICE with ARJ.
- Archive file reordering facility with the option of sorting by file size, file extension, CRC value, date-time modified, filename, pathname, compression ratio, file attribute, and more.
- String searching with context display within archive files.
- Built-in facility to recover files from broken archives.
- Self-extraction feature that is internal to the ARJ runfile. The SFX module is full-featured with a built-in help screen.
- Internal string data integrity check in ARJ to resist hacking à la LHARC to ICE.
- Archive security envelope feature to resist tampering with secured archives. This feature disallows ANY changes to a secured archive. Not even comments can be changed.
- Password option to encrypt archived files.
- Text mode data-compression option to enable movement of text files from one host machine to another. Text mode also results in slightly greater file size reduction on MS-DOS machines.
- File extraction to screen in a paged mode to permit browsing through an archive.
- Specification of the files to be added to an archive via one or more list files. In addition, ARJ can generate a list file.
- Specification of files to be excluded from processing by ARJ.
- Subdirectory recursion during compression and extraction.

## Release Notes

The only difference between the registered version and the shareware version is the version/copyright message. Registered users receive a registration code to change the version message.

The use of ARJ in a business, commercial, institutional, or government environment requires a license. However, business, commercial, institutional, and government users may use ARJ for evaluation purposes for a period of 30 days. See the file LICENSE.DOC for full details.

While evaluating ARJ, you should use the **-jt** (test archive) option to verify new ARJ archives of your data.

This version has been tested under DOS 2.11, 3.3, 4.01, and DOS 5.0 (and under 6.0 by Bob).

Here is a suggested command that will test ARJ on all of your files:

```
ARJ a -r -jt -y "-vasdel a:\vol.*"
   a:\vol c:\*.*
```

## ARJR, DEARJ, and REARJ Programs

The new programs ARJR and DEARJ are available to registered and licensed users of ARJ. ARJR is the ARJ program minus the help screen and SFX modules. DEARJ is the ARJR program minus the archive creation/modification functions.

There are two versions of REARJ, the shareware version and the registered version. The registered version contains a number of additional enhancements including environment variable usage, !listfile capability, and selection by date-time.

See the LICENSE.DOC and ORDERFRM.DOC files for more information.

UNARJ and DEARJ are *not* the same program.

## Quick Start to Using ARJ

Please note that switch options may be placed anywhere in the command line.

To create an ARJ archive containing all of the files in the current directory:

```
ARJ a archive
```

To create an ARJ archive containing all files with the **.DOC** extension in the current directory:

```
ARJ a archive *.DOC
```

To create an ARJ archive containing all of the files in the named directory and all files in subdirectories of the named directory:

```
ARJ a -r archive
   named_directory\*.*
```

To create an archive containing files without path specs:

```
ARJ a -e archive
   named_directory\*.*
```

For maximum compression, use the **-jm** or **-jm1** options. For better speed, use the **-m2** option:

```
ARJ a -r -jm archive
   named_directory\*.*

ARJ a -r -m2 archive
   named_directory\*.*
```

To create an ARJ archive containing the full specified pathnames of the stored files including any drive and root specs:

```
ARJ a -r -jf archive
   C:\top_directory\*.*
```

To back up your hard disk to multiple volume archives on drive A with archive testing and archive bit resetting:

```
ARJ a -r -jf -jt -a1 -b2 -vvas
   A:backup C:\*.*
```

To extract all of the files in an archive to the current directory:

```
ARJ e archive
```

To extract all of the files in an archive to a named directory:

```
ARJ e archive named_directory\
```

To extract all files with the **.DOC** extension to the current directory:

```
ARJ e archive *.DOC
```

To extract all of the files in an archive recreating the original directory structure:

```
ARJ x archive
   original_directory_name\
```

The ending \ character is optional if the original_directory_name already exists.

To extract all of the files in an archive containing absolute pathnames to the original paths:

```
ARJ x -jf archive
```

To list all of the files in an archive:

```
ARJ l archive
```

## How to Create an Executable Self-Extracting ARJ Archive

The command

```
ARJ y -je archive
```

will create a full featured self-extracting archive from an already built archive.

The command

```
ARJ y -jel archive
```

will create a smaller self-extracting archive.

Syntax: `ARJ y -je archive` produces archive.exe.

Under DOS systems other than 2.11, 3.2, 3.3, 4,0, and 5.0 you may have to rename the self-extract module to ARJSFX.EXE to do the extraction.

See the **–je** option for more information.

## Converting Other Archive Files to ARJ Format

Included with this software is the program REARJ. This program can be used to individually or collectively convert archive files from other formats to the ARJ format.

```
REARJ *.ZIP *.ARC *.LZH
```

will convert all ZIP, ARC, and LZH archives in the current directory to the ARJ format. See the REARJ.DOC for more information about REARJ.

## How to Use ARJ

If you type `ARJ [return]`, you will see a simple help screen.

If you type `ARJ -? [return]`, you will see more detailed help information.

## ARJ Limitations

ARJ will accept up to

64 filenames/wildnames on command line
16,000 filenames resulting from wildnames
8,000 filenames/wildnames to exclude
8,000 ARJ filenames resulting from wildnames
2,048 character comments
(up to 25 lines or 1 file)

For compressing, ARJ requires approximately 290,000 bytes plus the memory necessary to store all of the pathnames to be archived when using the default compression method **(–m1)**.

For extracting, ARJ requires approximately 166,000 bytes plus. The program DEARJ (available to registered users) requires approximately 123,000 bytes plus.

There is no limitation on the number of files that can be stored in one archive. However, each add command can only add a maximum of 16,000 files at a time depending upon memory availability. I expect that a normal maximum of 5,000 to 8,000 filenames can be handled without running out of memory during the compress phase.

If you do not have enough memory, you should use the **–l** switch to dump the filenames to a list file. You can then break the list file into smaller files and use multiple ARJ commands to archive all of the files.

**Example:**

```
ARJ a -r -lname.lst archive \*.*
```

If the above command fails due to lack of memory, split the name.lst file into smaller pieces named name1.lst, name2.lst, etc. Then execute:

```
ARJ a archive !name1.lst
ARJ a archive !name2.lst
.
.
```

ARJ currently does not see that wildnames like **C:\*.*** and **C:\*.*** can actually represent the same thing. ARJ would expand each of those two wildnames into a list that could be up to twice as long as necessary.

When updating an archive, ARJ creates a temporary file named ARJTEMP.$nn in the current directory or work directory. While ARJ is scanning a wildcard filespec, ARJ will change the name of the target archive to ARJTEMP.$nn while the scan is proceeding to avoid including the archive itself in an add or move command. Also, as a result, you cannot add a file named ARJTEMP.$nn to an ARJ archive. Please note that the name of this temporary file may change at a future revision of ARJ.

## Important Differences Between ARJ and LHARC

ARJ by default stores the full specified pathname of files archived minus any drive letter and root symbol. The **e** and **x** commands will by default extract all of the files in the archive without using date-time stamps to select files. You should specify **–u –y** to duplicate LHARC functionality.

The **f** command in ARJ requires the **–r** switch to be identical to the LHARC **f** command if the original archive was created using the **–r** switch.

## Important Notes

When using the **–w** working directory switch, ARJ does not check on space availability before overwriting the original archive if it exists. Be sure that you have enough disk space for the new archive before using the **–w** switch. If ARJ aborts in this situation because of disk space, ARJ will keep the temporary archive.

By default, ARJ does not see hidden or system files. ARJ will process system and hidden files when you specify the **–a** switch.

Like LHARC and PKZIP, ARJ requires extra disk space to UPDATE an archive file. ARJ will back up the original archive while it creates the new archive, so enough room must be available for both archives at the same time.

Currently, ARJ will not extract overwriting a read only file.

## Tips to Using ARJ Efficiently

When archiving to diskettes, you should use the **–w** option to set a working directory on your RAMDRIVE or hard disk drive to speed up building the archive. You should use the **–jt** option when archiving to diskettes or when you really want to be sure that ARJ will be able to extract what you have archived. There are cases where your hardware or memory-resident software will corrupt your work, so the **–jt** option is excellent insurance.

Using the **–js** option saves time by not compressing archives.

You should use the **–e** option whenever you do not need to store pathnames in an archive that you are creating. This will save space.

Convert an ARJ archive into a self-extracting archive with a command like the following:

```
ARJ y archive -je
```

To capture a comment from an ARJ archive, use the following command:

```
ARJ e archive ... -zcomment.txt
```

ARJ has several compression methods that provide size/time tradeoffs. Method 4 **–m4** is about twice as fast as method 1. The **–jm1** and **–jm** options modify the **–m1** and **–m2** options to provide even greater compression at a cost in time.

## Using ARJ Within Other Programs

Since ARJ uses over 280,000 bytes of memory during compression, it is difficult to use ARJ in a large application program unless that program swaps itself out of memory when it executes DOS commands like ARJ. However, there is at least one shareware program available that will automatically swap your large application program out of memory whenever it shells out to DOS to execute a command. The program SHROOM by Davis Augustine should be able to solve this memory problem for you. The latest version is named SHROM18J.ZIP on Channel One BBS. According to the SHROOM documentation, you can reach the author at:

CompuServe id 72230,3053
Davis Augustine, P.O. Box 390178,
Cambridge, MA 02139

This is not an endorsement of the product SHROOM.

The easiest way I have found to use this product is to type

```
SHROOM COMMAND
```

SHROOM –v COMMAND will let you see SHROOM in action when you shell out to execute a DOS command.

## ARJMENU Program

A new program called ARJMENU by Michael McCombs will be released shortly. As far as I know, it is the only menu-driven interface program that supports ALL of the features of ARJ. This program is aimed at users who hate command-line interfaces. ARJMENU allows the user to pick and choose ARJ options. The user does not have to remember the ARJ switch syntax. The newest version of the program supports ARJ version 2.30. You can reach the author at:

Internet/ARPANet:
mccombs@sumax.seattleu.edu
Michael McCombs, 517 Ninth Ave. #310,
Seattle, WA. 98104

## Using ARJ as a Backup Program

ARJ can be used as a substitute for a backup program. However, it does not have the diskette critical-error handling or data recovery facilities of a FASTBACK, etc. So you should be sure of the reliability of your diskettes.

The following partial command lines illustrate a full backup command, an incremental update command, and a restore command. The only parts missing are the names of the files to backup/restore.

```
ARJ a A:backup -r -vvas -a1 -b2 -i1
  -js -jt -jiC:\backup.inx -wC:\ -m4
```

```
ARJ a A:backup -r -vvas -a1 -b1 -i1
  -js -jt -jiC:\backup.inx -wC:\ -m4
```

```
ARJ x A:backup -vv -jycn
```

You should familiarize yourself with the above switches so that you can modify the above command lines as needed.

If you have a RAMDRIVE large enough, you should change the **–w** option to point to the RAMDRIVE.

If you have enough free hard disk space, you can build all of the diskette volumes on the hard disk for later copying to diskette. In this case, you will need to change the name of the archive to **C:backup** or similar. The **–vvas** option should be changed to **–v360**, **–v720**, or whatever is appropriate for your diskette size.

Please note that 360, 720, 1200, and 1440 are abbreviations for the standard diskette sizes. Other sizes will require your entering the entire number. Another change is to add the option **–y** which will turn off the `Ok to proceed ...` prompt. Lastly, if the **–w** option is pointing to the hard disk, you should remove the **–w** option entirely.

```
ARJ a C:backup -r -v360 <other
options> -m4 -y
```

**Important:** Only a maximum of 100 volumes can be built on a disk at one time because of the volume suffix rolling over at *.A99 to *.A00 when using default archive naming. However, if you specify the starting archive name with a suffix of **.000** or **.001**, ARJ will create 1,000 or 999 uniquely named volumes.

```
ARJ a C:backup.000 -r ..........
```

Both backup commands will pause for a system command. You can execute DOS commands at this point. This is a suitable place to do a **dir a:** to make sure that your disk is formatted and has enough free space on it. You may need to execute **format a:** or **del a:\**. A very useful command might be **QDR A:**. QDR is a utility from Vern Buerg. You will need to type **exit** to allow ARJ to continue.

If the backup fails after completing one or more diskettes, you can restart at the next archive after the last successful volume. You will need to examine the information in the BACKUP.INX file to find the name of the file that is to start this archive. It will usually be the same as the last filename in the previous volume. You will also need the byte position to start in this same file. That can be determined from the information in BACKUP.INX. You can then retype the exact same backup command as before with a few changes. You will append the right **.Ann** suffix to the archive name, and you will add the options **–jx** and **–jn** with the proper arguments.

For example, if the above full backup command failed during diskette two on filename DOS\MODE.COM which was started at byte 125, this would be the correct command:

```
ARJ a A:backup.A01 -r -vvas -a1
<other> -m4 -jx125 -jnDOS\MODE.COM
```

The most error prone step is determining the correct **–jn** option. Be sure to spell the filename exactly the same as it appears in the BACKUP.INX file. A common error concerns the use of the root directory symbol \ with the **–jn** option. Verify the presence or absence of the root symbol \ in the backup index file.

If the restore fails after one or more diskettes, simply retype the same command as before but add the right **.Ann** suffix to the archive name. If ARJ has aborted because of a disk full on a file split between volumes, you will have to restart at the first volume that contains that file.

## Backup-Type Archives and Backup Files

New to ARJ 2.30 and ARJ archives is the concept of backup-type archives. Normally, when adding duplicate files to an ARJ archive, ARJ will overwrite the existing file. However, at ARJ 2.30 with a backup-type archive, ARJ will keep the existing file by marking it as a backup and inserting the new file. This will make ARJ archives larger than normal. It is possible to have more than one duplicate backup file. You can make an archive a backup type by using the **–jb** option during the add or by typing

```
ARJ y archive -jb
```

Any further additions of already existing files will result in more files marked as backups. You may reset that backup flag by specifying the **–jb1** option as in

```
ARJ y archive -jb1
```

This feature is useful when backing up frequently modified files. One archive can have several days worth of backups. The existence of backup files or the backup archive flag is indicated when executing the **l** or **v** commands. There will be a message at the end of the display. The backup files themselves can be displayed by using the **–jg** or **–jg1** options with the **l** or **v** commands.

To extract a specific backup file, you have several options. You can execute

```
ARJ e archive filename -jg -q
```

and enter **yes** on the specific instance that you wanted extracted. You can specify the exact modification date-time that you want as in

```
ARJ e archive filename -jg -
o910101120000 -ob910101130000
```

You can specify the sequence number of the file in the archive. To extract the fifth file in the archive, type

```
ARJ e archive 5 -jg -#
```

The sequence number is available in the **v** command display. Or you can extract all occurrences of the file with

```
ARJ e archive filename -jg -jo
```

Duplicates will be renamed with numeric file extensions (.000).

The **ARJ k archive *.*** command can be used to purge an archive of all backup files.

You can unmark backup files with **ARJ y archive –jb2**. You can unmark specific files such as the fifth file in the archive with

```
ARJ y archive -# -jb2 5
```

You should not use the **–e** option when adding files to a backup-type archive because ARJ will not have the ability to distinguish between files with the same filespec and different pathspecs. You could end up with duplicate files in the archive as opposed to one recent file and several backup files.

## The Filespec "..."

Several times in this document and the UPDATE.DOC file, there is mention of the filespec "...". This filespec is chosen so as to not match any existing filename. ARJ will *not* generate an error or warning for not matching "..." specifically.

## ARJ Error Situations

The following is a description of how ARJ will respond in various error situations.

### Add:

If a user-specified file is not found during an add, ARJ will continue processing and will keep the archive and terminate with an error condition. In a disk full condition or any other file i/o error, ARJ will promptly terminate with an error condition and delete the temporary archive file unless the user has specified the **–jk** switch.

### Move:

ARJ will only delete files that have been successfully added to the archive. If you have specified the **-jt** (test) switch, ARJ will abort on any error. If you specify the **-jk** switch, ARJ will not delete the temporary archive upon an abort.

### Extract:

In a disk full condition or any other file i/o error, ARJ will promptly terminate with an error condition and delete the current output file.

### CRC errors or bad file data:

In the case where an ARJ archive has been corrupted, ARJ will report a CRC error or a Bad file data error. These corruptions can be the result of an unreliable diskette, a computer memory problem, a file transfer glitch, or incompatible CACHING software. Most of these errors are the result of file transfer glitches and bad diskettes. A few are the result of an incompatible interaction with SUPER PCKWIK 3.3 advanced diskette support or Windows 3.0.

**ARJ DOS errorlevels and their meanings:**

| | |
|---|---|
| 0 | success |
| 1 | warning (specified file to add to archive not found, specified file to list, extract, etc., not found, or answering negatively to `OK to proceed to next volume...` prompt) |
| 2 | fatal error |
| 3 | CRC error (header or file CRC error) |
| 4 | ARJ-SECURITY error or attempt to update an ARJ-SECURED archive |
| 5 | disk full or write error |
| 6 | can't open archive or file |
| 7 | simple user error (bad parameters) |
| 8 | not enough memory |

## ARJ User Action Prompts

ARJ prompts the user for action at certain times. There are several types of prompts. One is for yes/no permission, another is for a new filename, another is for archive comments, and one other is for search strings. The yes/no prompts will also accept **quit** for program termination and **always** to bypass further user prompts. Since ARJ uses STDIN for user input, be careful about typing ahead, anticipating prompts. ARJ may prompt you for an unexpected action and use your earlier input. The **-jy** option lets you change the prompting modes to single character query mode. See the section on **-jy** for more information.

## ARJ Environment Variable

ARJ will first look for an environment variable named ARJ_SW and use its value as switch options for ARJ. If ARJ finds such an environment variable, it will display a message to that effect.

You can inhibit ARJ from using this environment variable by using the -+ option. You can also set the name of the environment variable with the -+ option. For example, to instruct ARJ to use the switches set in the environment variable ARJ_SW2, you might use a command such as the following:

```
ARJ a -+ARJ_SW2 archive
```

### Specifying environment switches

Use the following syntax to individually specify default ARJ switches:

```
SET ARJ_SW=<switches>
```

**Example:**

```
SET ARJ_SW=-w\temp -k -e
```

Do *not* add any blanks after the variable name ARJ_SW. As in LHARC, command-line switches can be selected to override ARJ_SW settings. ARJ will allow you to use a different switch character - or / in ARJ_SW and in the command line except when using the **-ju** (unix) option.

### Specifying switches in a configuration file

If the ARJ_SW environment variable specifies a filename (text not beginning with a switch character), ARJ will open that filename and scan it looking for a line of text that begins in column 1 with the same letter as the ARJ command being executed. The following text is processed as the ARJ_SW switches. This allows each ARJ command to have its own switch settings. In addition, ARJ will also look for the + (plus) symbol in column 1 to use as a miscellaneous switch string. This string will be added to any command switch string found (if any). If no command switch string is found, ARJ will use the miscellaneous switch string as the ARJ_SW switch settings. Note that the + switch settings will *not* be used if the + string occurs below the command switch string.

For example, to instruct ARJ to use the switches in the file ARJ.CFG, you can use the following command:

```
SET ARJ_SW=C:\ARJ\ARJ.CFG
```

where C:\ARJ\ARJ.CFG contains

```
a -jm1 -jt -i1
c -zcomment.txt
+ -jv
l -jp
e -i1
```

AND is equivalent to

```
a -jm1 -jt -i1
c -zcomment.txt
l -jp -jv
e -i1 -jv
f -jv
v -jv
all other commands
.
.
.
```

In the above example, any ARJ **a** commands will use **-jm1 -jt -i1** as the ARJ_SW switch options.

The ARJ_SW variable or the ARJ_SW configuration file switch settings may *not* have quoted switches such as **-vasformat a:**.

## ARJ Command-Line Syntax

```
ARJ <command> [-<switch>[-|+|
   <option>]...]
   <archive_name>[.ARJ]
   [<base_directory_name>\]
   [<!list_name>|<path_name>|<wild_name>...]
```

Commands and switches can be entered in upper- or lowercase. Switches can be placed anywhere after the command ARJ.

ARJ supports the use of either – or / as the switch option character. The first occurrence of either – or / that ARJ encounters in the ARJ_SW variable will determine the switch symbol for processing ARJ_SW. The first occurrence of either – or / that ARJ encounters in the command line will determine the switch symbol for processing the command line except when the **-ju** (unix) option has been specified in ARJ_SW. You may *not* mix and match switch symbols.

Throughout this document, the symbol / may be substituted for – in switch usage.

**Examples:**

```
ARJ a A:archive *.* /va /r
```

is correct.

```
ARJ a A:archive *.* /va -r
```

IS INCORRECT USAGE!

Switches specified on the command line will either toggle or override switches specified with the ARJ_SW environment variable. Switch usage is identical to that of LHARC.

| | |
|---|---|
| `-s+` | turns on switch **s**. |
| `-s-` | turns off switch **s**. |
| `-s` | toggles the state of switch **s**. |
| `-sname` | provides the name argument for switch **-s**. |
| `--` | skip processing of any more switch options. |

Switch options CAN be combined to save command-line length. However, some switch options take optional string arguments and, therefore, must be the last option in a combined argument token string because ARJ accepts the rest of the argument token as the optional argument. You can combine **-r** and **-i** to make **-ri**. You can combine **-wd:** and **-i** with **-iwd:** because the switch option taking the string argument is last. Please note that switches that *only* accept the +, –, **0**, **1**, **2** modifiers can be combined in any order. The following switches must be last in a combined switch argument: **-g**, **-l**, **-m**, **-o**, **-t**, **-v**, **-w**, **-x**, **-z**, **-!**, **-$**, **-jc**, **-jd**, **-jh**, **-ji**, **-jn**, **-jp**, **-js**, **-jw**, **-jx**, **-jy**, **-jz**.

***Warning:*** The **j** in **-jX** option switches is a SHIFT symbol. Therefore, a combined **-jatv** is a combination of **-ja**, **-jt**, and **-jv**, since the **j** shifts the meaning of subsequent symbols in the argument token string. The use of more than *one* **j** shift symbol per argument token string is *not* supported for future versions.

**Examples:**

```
-rijvta <=> -r -i -jv -jt -ja
-i1kwd:\ <=> -i1 -k -wd:\
```

ARJ will accept an ending – on most switches such as **-w-**. The **-x** switch option is one exception. The one caveat is that for switches that support an optional argument string such as **-wd:**, that switch option must *not* be followed by a concatenated switch option such as **–w–r**. An argument token string such is **–rikw–** is acceptable because the **–w** option is the last one. One special switch combination is the **–jyyr** combination. You can turn off the **y** and **r** modifiers with **–jyyr–**.

The switch option -- tells ARJ that there are no more switch options to process in the current command line. This is useful when you need to enter filenames beginning with –.

**Example:**

```
ARJ a archive-----testfile
```

The standard ARJ file suffix is **.ARJ**. Subsequent multiple volume archives end in **.A01**, **.A02**, etc, up to .A99, .A00, .A01. It is possible to have multiple volumes start at **.001** and go up to **.999**.

The ARJ command must be the first nonswitch argument after **ARJ**. The ARJ archive name must be the first filename on the command line. The base directory, if any, must be the second filename argument. The switches and other filenames can be in any order. The base directory name should end with \ (backslash) or : (colon). However, ARJ will still accept directory names without the \ character if the directory already exists. This feature is limited to the add-type and extract-type commands.

You can specify a wildcard for the archive name such as ***.ARJ** for all ARJ commands except for the add commands (a, f, j, m, u). If you also specify the **–r** switch, ARJ will search subdirectories for ARJ archives (*.ARJ) also.

**Examples:**

`ARJ l -r *` will list all of your *.ARJ files.

`ARJ c *.arj -zbbs.cmt` will comment all of your archives.

Wild_names follow MS-DOS convention. ***.*** means all files. ***.DOC** means all files with an extension of .DOC. **?B*.*** means all files with a second character of **B**.

The default for <wild_name> for all commands except for **d** is ***.***.

For the add, move, freshen, and update commands, filename matching in the archive requires an exact path match depending upon the **–e** option. For non-update commands, specified filenames with paths will force a full pathname match.

You can supply one or more filenames for files containing lists of files to be added to an archive. The filenames must be listed one per line with no leading or trailing blanks. The list filename(s) must be prefixed with **!**. If you want to archive a filename beginning with **!**, you must use the **–!** option to set a new list file character.

You can exclude filenames/wildnames from the list of filenames to be processed by ARJ.

**Example:**

```
ARJ a software *.* -x*.exe -x*.obj
```

adds all files in the current directory except .EXE and .OBJ files.

***Note:*** See the ARJ.DOC file for the detailed command reference omitted here.

## Sample ARJ Commands

### a Add files to archive

ARJ a archive
ARJ a archive –n
ARJ a archive –f
ARJ a archive –jt
ARJ a archive –jt1
ARJ a archive –jt2
ARJ a archive –q
ARJ a archive –r
ARJ a archive dir\ –o
ARJ a archive dir\
ARJ a archive dir\ *.doc *.exe
ARJ a archive !names.lst
ARJ a archive --name–.txt
ARJ a archive name1 name2 name3
ARJ a a:archive –we:\ –vas c:\ –r –b2
ARJ a a:archive.001 –we:\ –vvw360 c:\ –r –b1
ARJ a archive –vw360 c:\ –r –y
ARJ a archive c:\ *.* –r –jt1 –jf

### b execute Batch or DOS command

ARJ b archive name.txt
ARJ b archive –jwOUT.NAM
ARJ b archive SHEZ.EXE –jwSHEZ.EXE
ARJ b archive *.exe –jwTEMP.EXE "–jq
TEMP –help"

### c Comment archive files

```
ARJ c archive
ARJ c archive –z
ARJ c archive –zcomment.txt
ARJ c archive –zNUL
ARJ c archive file1 –jzcomment.txt
ARJ c archive *.doc –jzNUL
```

### d Delete files from archive

```
ARJ d archive *.doc
ARJ d archive *.* –jg
ARJ d archive *.* –n
ARJ d archive *.* –y
ARJ d archive !names.lst
```

### e Extract files from archive

```
ARJ e archive *.doc
ARJ e archive *.doc –d
ARJ e archive –jd50000
ARJ e archive –jo
ARJ e archive –c
ARJ e archive dir\ *.doc –jt
ARJ e archive *.doc –y
ARJ e archive –# 10 15 20–40
```

### f Freshen files in archive

```
ARJ f archive
ARJ f archive –r
ARJ f archive dir\ –r
ARJ f archive dir\ *.doc –r
ARJ f archive dir\ –r –jt2
```

### g Garble files in archive

```
ARJ g archive –gpassword
ARJ g archive *.doc –gpassword
ARJ g archive –g?
```

### i check Integrity of ARJ.EXE

```
ARJ i
ARJ i c:\bin\arj.exe
```

### j Join archives to archive

```
ARJ j archive name1.arj name2.arj name3.arj
ARJ j archive *.arj
ARJ j archive arjsfx.exe
ARJ j archive *.arj –o911201
ARJ j archive *.arj –r
```

### k pacK bacKup files in archive

```
ARJ k archive *.*
ARJ k archive *.* –y
ARJ k archive *.doc –n
```

### l List contents of archive

```
ARJ l archive
ARJ l archive *.doc
ARJ l archive –o911201
ARJ l archive –o911201 –ob911231
ARJ l archive –n
ARJ l archive –jg
ARJ l archive –jp
```

### m Move files to archive

```
ARJ m archive
ARJ m archive *.doc
ARJ m archive dir\ *.doc
ARJ m archive dir\ *.doc –o
ARJ m a:archive.001 dir\ –r –vvwas –jt2
```

### n reName files in archive

```
ARJ n archive
ARJ n archive *.doc
ARJ n archive !names.lst
ARJ n archive –o911201
```

### o Order files in archive

```
ARJ o archive !names.lst
ARJ o archive *.bat *.exe *.doc
ARJ o archive file1 file2 file3
```

### p Print files to standard output

```
ARJ p archive
ARJ p archive –jp
ARJ p archive *.doc
ARJ p archive –o911201
ARJ p archive name.txt > PRN
```

### r Remove paths from filenames

```
ARJ r archive
ARJ r archive *.doc
ARJ r archive –o911201 –ob911231
```

### s Sample files to screen with pause

```
ARJ s archive
ARJ s archive *.doc –y
ARJ s *.arj –y –r
```

### t Test integrity of archive

ARJ t archive
ARJ t archive *.exe
ARJ t archive !names.lst
ARJ t archive –jt1

### u Update files to archive

ARJ u archive *.doc
ARJ u archive *.doc –jo

### v Verbosely list contents of archive

ARJ v archive
ARJ v archive *.doc
ARJ v archive –jv
ARJ v archive –jv1
ARJ v archive –jp
ARJ v archive –jg

### w Where are text strings in archive

ARJ w archive
ARJ w archive –jp
ARJ w *.arj –jp
ARJ w *.arj *.doc
ARJ w *.arj *.txt –jq+5+string
ARJ w archive "–jq–3–to be or not"

### x eXtract files with full pathname

ARJ x archive *.doc
ARJ x archive *.doc –d
ARJ x archive –jd50000
ARJ x archive –jo
ARJ x archive –c
ARJ x archive dir\ *.doc
ARJ x archive –jf
ARJ x archive *.doc –jyc

### y copY archive with new options

ARJ y archive –je
ARJ y archive –je1
ARJ y archive –jb
ARJ y archive –jb1
ARJ y archive –jb2
ARJ y archive *.bat –jb2

## Known ARJ Issues/Problems

Due to the inner workings of Turbo C++, ARJ may run out of memory if your computer is configured with thousands of bytes of environment variables. A workaround is to spawn another COMMAND.COM as in

```
command /e:200 /c arj x archive
.........
```

When using a working directory, ARJ does not check for disk space before overwriting the original archive. Be sure you have enough space before updating an archive using the **–w** switch.

Because of a bug with extended archive header processing, you should convert to ARJ 2.10 and above as soon as it is practical to do so. This bug is *not* a current problem because no version of ARJ supports the use of the extended header.

TSRs that activate via *hot keys* may be inhibited by ARJ during user keyboard input. This is a "feature" of the Turbo C++ getc() function. A partial workaround is to use the ARJ **–jyry** option to go to single key query mode.

A similar problem occurs with HYPERDISK and staged writes. HYPERDISK's timer delay appears to be inhibited during ARJ user keyboard input such as during `Ok to ... volume/diskette?`. This causes an error when you swap to the next diskette. The workaround is to use the ARJ **–jyry** option to go to single key query mode.

There is a reported problem using ARJ and floppy disk drives when using the disk cache program SUPER PCKWIK 3.30 with advanced diskette support (/D+). The system may occasionally hang when ARJ attempts to read/write the diskette. You may also get CRC or Bad file data errors. Disabling the SUPER PCKWIK option with /D– appears to remove this problem. Other programs have similar problems with this feature of SUPER PCKWIK.

***Note:*** Please see the ARJ.DOC file for information on technical support, acknowledgments, ARJ availability, distributors, etc.

## ARJ Registration Form (rev 7):

Name: ____________________________
Company: __________________________
Address: ___________________________
City: _____________________________
State: __________ Country: ______________
Zip: ___________ Phone: _______________

**Remit to:**

Robert K. Jung
2606 Village Road West
Norwood, Massachusetts 02062 USA

Prices subject to change without notice.

INDIVIDUAL REGISTRATION for the use of ARJ, REARJ, ARJSFX, and ARJSFXJR for MS-DOS:
each registration is $35 ........................ $ ______

ARJ-SECURITY ENVELOPE
serial number(s) $50 each .................... $ ______

Please note that this is a separately charged item. List the exact name(s) that you want displayed by ARJ security envelope. The display will look like the following line:

ARJ archive created by (this is where your name is placed) R#nnnn

______________________________________

(Exact name, MAX of 60 characters, add more sheets if necessary)

When ordering software diskettes, licensees and registrations get the REARJ, ARJR, and DEARJ programs in addition to the normal ARJ software.

Current software & documentation
diskette(s) .............................$5 each $ ______

Next ARJ release software &
docs diskette(s) ...................$5 each $ ______

TOTAL PRODUCT .................................. $ ______

Massachusetts residents add
MASS sales tax ............................5% $ ______

For orders outside the USA, add
shipping and handling .............$5.00 $ ______

TOTAL ORDER ........................................ $ ______

For payment by non-USA BANK CHECK,
add (see TERMS below) .............$40.00 $ ______

TOTAL ENCLOSED ................................ $ ______

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Payment: ☐ Check ☐ Cash ☐ Money order
☐ Credit card - See instructions in file CREDIT.CRD

Authorized Signature:

______________________________________

TITLE: ________________________________

DATE: _________________________________

TERMS:

Payment must be made by check, US POSTAL Money Order, or International Money Order in USA funds drawn on a USA bank made out to Robert K. Jung. Corporate purchase orders (net 30) for orders over $200 will be accepted from FORTUNE 500 corporations within the USA and Canada. All orders outside of the United States and Canada must be prepaid. Checks may require two weeks to clear. Please allow a few weeks for delivery. Credit card orders for ARJ software are handled by arrangement with Public Software Library. See the file CREDIT.CRD for more information. If necessary, CHECKS drawn on a non-USA bank for amounts over $100 (USA) or its equivalent will be accepted. Please add a $40 (USA) surcharge to cover handling, currency exchange, and bank transfer costs. This surcharge does NOT apply to bank checks which are written for USA funds with a USA bank named on the check. Please allow several weeks for the check to clear and for delivery.

# BATUTIL

**Version 4.0**
**Copyright © 1988–1991 by Barry Simon and Richard Wilson**
**CTRLALT Associates (R)**

## Chapter I: Overview

### Introduction

BATUTIL is a program included in the STACKEY package by CTRLALT Associates. It can only be registered as part of STACKEY.

***Note:*** The documentation for BATUTIL is quite extensive. Only a small portion of it is duplicated here (in *DOS 6 SECRETS*). Please see the BATUTIL.DOC file (and the other files included on the diskettes) for more detail on how to use BATUTIL.

### I.2 What BATUTIL Does

BATUTIL is a program with two purposes: to give you power inside your batch files and to give you more control over the DOS environment. To get a feel for the program, please run BUDEMO.BAT which is included with the distribution files.

Included in the information that you can get returned in either the DOS errorlevel, stored in BATUTIL's internal variables, or an environmental variable are

- Current time, date, day of the week.
- Total amount and amount free of disk space, memory, and EMS.
- CPU type and type of coprocessor if present.
- Whether a file exists not only in the current directory but also on the DOS path. If it exists on the DOS path, the actual directory can be returned in another environmental variable.
- Whether a file has today's date or not.
- Whether one of two files is older than another.
- You can parse a filespec into individual components.
- You can do arithmetic on integers and dates.
- You can manipulate strings including case change and centering.
- Whether Desqview or Carousel is running and, if so, which partition you are in; whether Windows is running underneath you.

Some of these options may not seem so useful at first sight, but, for example, whether a file has today's date or not can be used to make a routine that will only get run once per day. Chapter VII will explain sample uses of BATUTIL.

Central to the design of BATUTIL is the notion of "high level" language. There are much more sophisticated batch languages available, and with them one can write menus at least as involved as are available in BATUTIL's MENU command. But doing so requires you to write a little program in the batch language while MENU is a built-in command of BATUTIL; if you want to, BATUTIL comes with a full-featured language dubbed BUSIC. It includes IF..THEN..ELSE, CASE, WHILE, REPEAT, and GOTO, all of which can be used in the style of DOS batch languages.

In addition to getting information from the operating system, you can pass information to and get information from the user. BATUTIL gives you considerable control over displaying information including the possibility of turning the cursor on and off and displaying messages in various colors. And BATUTIL understands the metastrings that STACKEY does, so you can easily display data such as today's date in English.

User input can be obtained in various forms. A getkey routine lets you list a set of keys using STACKEY syntax and get which key in the list the user has hit:

- Whether a lock key is currently pressed.
- With some simple commands, you can pop up an elegant menu for the user to pick from and have the choice returned in the errorlevel.

- You can have the user input a string and get it stored in the environment.
- You can have the user type in a user name or password and see whether it matches a predetermined list and have which item is matched reported in the errorlevel.
- You can pop up a filename list for the user to choose from and have the answer stored in the environment.

BATUTIL also gives you considerable control over the DOS environment. It is able to do this by using undocumented features of the operating system. As always, such features should be used with care.

**Important:** Please read carefully the warning at the start of Chapter V concerning using undocumented features to access the DOS environment.

BATUTIL's environmental control has been tested with PC DOS Version 2.0, 2.1, 3.0, 3.1, 3.2, 3.3, 4.0, 5.0, and some flavors of MS-DOS. Besides being able to place information into the environment via user input, BATUTIL will

- Display more information about the environment than the SET command.
- Allow you to put strings in the environment up to a length of 255 characters rather than the 127 character maximum that DOS allows. In particular, with BATUTIL your PATH string can be up to 250 characters rather than the 122 characters that you can enter with DOS. (While DOS doesn't care about long paths, you may have an application program that does and crashes if it finds a path over 127 bytes.)
- Allow simple commands to add a directory to your PATH without inadvertently adding one already there and allow deletion of a directory from your PATH.
- Allow full screen editing of your environment and PATH.
- Allow you to save the environment to a file, to load or merge a file into the environment, and to kill the environment (use with care).

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the BATUTIL files in the directory \SECRETS\BATUTILS. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the BATUTIL files to a directory that is included in the search path.

**The files in the BATUTIL package are as follows:**

| | |
|---|---|
| BATUTIL.EXE | The basic program. The only file required to use BATUTIL. |
| BATUTIL.HLP | Online help called by BATUTIL, if you use BATUTIL ? or BATUTIL !. |
| BATUTIL.DOC | Documentation in electronic form. |
| BUDEMO.BAT | Sample batch file which will call the other sample batch files. |
| COLORDEM.BAT | Sample batch file showing color capability. |
| EQUIP.BAT | Sample batch file showing ability to read hardware. |
| INPUTDEM.BAT | Sample batch file showing ability to get input from the user. |
| MENUDEMO.BAT | Sample batch file showing BATUTIL's menu-making capability. |
| SHOWDEMO.BAT | Sample batch file showing display options including big characters and metastrings. |
| SOUNDEMO.BAT | Sample batch file with BATUTIL's 20 sounds/tunes. |
| LANGDEMO.BAT | Sample batch file demonstrating the new language features of BATUTIL 4.0. |

| | |
|---|---|
| WHATEL.EXE | Program described in Section II.7 to determine errorlevel and timing for some commands. |

Because of our desire to keep the basic files on a single disk and to keep download time low, we are encouraging distribution of a small shareware version without the BATUTIL.HLP file. If you have a version without this file and want it for evaluation, it can be downloaded from our support section of CompuServe as BUHELP.EXE (Section 12 of PCVENA), or you can get it from many disk vendors including PBS (1-800-426-3475). If you upload this program to a BBS, please keep BATUTIL.HLP as a separate file.

***Important:*** Users of 4DOS Version 2.x, please read Section V.1 before calling for technical support. See Section I.4 for items in Version 1 not carried over to Version 4.x.

## I.3 Help, Verbose Modes, and European Dates

You can obtain interactive help for BATUTIL by typing in

```
batutil ?
```

or

```
batutil !
```

at the DOS prompt. To obtain help, the binary file BATUTIL.HLP must be available in the current directory, the directory where BATUTIL is loaded, or in your path. BATUTIL looks for the first two nonblank characters following the ? (or !), and if they are a valid BATUTIL command, you will be immediately sent to the help for that command. Otherwise, the main help menu will be displayed. For example

```
batutil ? SE
```

would display help for BATUTIL's SET command as would

```
batutil ?see the beautiful program
```

BATUTIL ? displays the menu-oriented help with various special effects. If these effects and the slightly longer time they take bother you, **batutil !** will display help without the special fades for displaying material.

The help program uses color attributes on a graphics monitor. If you have a two-color graphics monitor (such as on a laptop) or the colors are hard to see, the help program will use monochrome attributes if you use **SET BU!=bw** in your AUTOEXEC.BAT file or at the DOS command line before running BATUTIL. Note that on a true monochrome adapter (IBM MDA or Hercules monographics card), the help program will not use color attributes whether you use that set command or not.

Normally, BATUTIL exits when there is an error and sets the errorlevel between 200 and 253. There are three levels of visual error reporting that you can have turned on. For debugging purposes, the default is a verbose mode where BATUTIL will explain why it is exiting. For batch files you distribute, you may want to have verbose mode turned off and instead use a Quiet mode where no messages are displayed. To turn on this quiet mode, just make the first nonblank character in the command line the letter Q. Thus,

```
batutil {}
```

would display an error message, while

```
batutil Q {}
```

would exit without any message. For distributed batch files made with BATUTIL, you may want to turn on quiet mode using an environmental variable as described in Section I.5. Command-line mode takes precedence over any mode set in the environment and, in particular, you can override the mode set in the environment and restore to the default Verbose mode by making V the first letter on the command line after BATUTIL.

Chapter IX of the BATUTIL.DOC file has a complete list of error codes with an explanation of each. The on-line help main menu has an option to list all error numbers and their meaning.

In batch files, the error message may disappear too fast for you to see. In addition to Verbose mode, BATUTIL has a Pause mode which shows the error message and waits for a key before exiting. Pause mode is set with the letter P.

To see the information which BATUTIL is placing in the environment while testing out what a command will do, the following one line batch file (or an equivalent CED synonym), is useful:

```
batutil %1 %2 %3 %4 %5 %6 %7 %8 %9
  {EC $x(rc)}
```

The final command echoes the current value of the environmental variable RC to the screen.

There is an additional special parameter you can use at the start of the command line: E for European. It will change to European date conventions; so, while

```
batutil {ec $E}
```

would echo

```
December 19, 1991
```

the command

```
batutil E {ec $E}
```

would echo

```
19 December 1991
```

The E command affects the following: the meaning of $d, $E, and the translation used for dates. For example

```
batutil {ec $E(1/2/91)}
```

returns

```
January 2, 1991
```

while

```
batutil E {ec $E(1/2/91)}
```

returns

```
1 February 1991
```

since 1/2/91 is also translated using European conventions.

## I.4 New in Version 4.0

Version 1.0 was released bundled with Stackey 3.0 and this numbering caused some confusion; so, with this upgrade, we have upped the version number of BATUTIL all the way to 4.0 to make the numbers the same.

The most significant difference in this new version is the addition of a built-in language described further in Sections I.9, I.10, and Chapter VI. For this language to work effectively, we needed to remove the restriction of a single 127-character command line, and it is now possible to read commands in from a file as described in Section I.7. As a bonus, you need not load BATUTIL many times and that increases the speed of command processing by a considerable amount.

This may mean that you'd like to use BATUTIL's RUn command to run a program from a BATUTIL batch file without exiting BATUTIL. In Version 1.0, BATUTIL took over 200K when you used RUn, limiting it for these purposes. With this version, BATUTIL will swap itself to the hard disk or EMS memory leaving a core of about 11K in conventional RAM. About 300K of disk space or EMS is required. By default, EMS is used if available, but you can tell BATUTIL to use the disk. This swapping mechanism RUn command will not work on systems with floppy drives only and no EMS; so, you'll have to use the non-swapping mode (set up by placing **BUSWAP=!** in your environment).

**Included in the language support are the following:**

- Labels and Goto
- Calling subroutines
- IF..THEN..ELSE and CASE
- WHILE and REPEAT
- 32 internal variables as well as the use of the DOS environment for storing information
- FOR loops using an internal loop variable
- FOR loops using a list
- FOR loops using file wildcards like DOS batch files
- FOR loops varying through a directory tree
- An interactive trace mode
- String manipulation including change of case, length, formatting, searching, and word count
- Integer arithmetic with four functions, mod, and power
- Date manipulation including date arithmetic
- Reading and writing to files with optional metastring translation
- Reading from the screen
- Memory peeks and pokes
- I/O port peeks and pokes

**Other new features include the following:**

- Reading the number of file handles and DOS buffers
- Determining if BATUTIL is running in a Windows DOS box
- Determining if Norton's NDOS is running
- Determining if SHARE is loaded
- Determining the value of last drive
- A command to display text boxes with choice of frame characters and shadows
- Mouse support in the $F routines
- A $N metastring like $Q but with input restricted to numbers and a test to see if a string is a number between allowed maximum and minimum values
- The #I command which now identifies 486 CPUs
- Super metastrings to force commands that do not do metastring translation on their parameters to do such translation after all
- Menu which can now be set to timeout after a fixed time
- Support for European dates
- A password mode for entry of strings which are masked

*Note:* Please see the BATUTIL.DOC file for a list of changes from Version 1.0.

## I.5 Basic Syntax

**Command:** `REmark`

You give BATUTIL a series of commands on the command line or read them in from a file. There are about 130 different commands. Lest that overwhelm you, we note that many commands are just giving different pieces of hardware information which you'll rarely want. Even the most complex idea (menus) is controlled by one basic command and by fewer than ten commands which effect options (like whether the menu explodes). Chapter VIII is a detailed alphabetical command reference. When you invoke BATUTIL's help, one option is an alphabetical list of commands which will give you a panel with syntax and parameters for each.

The basic syntax is

```
batutil {command1} {command2} ...
```

Each command is placed inside braces {} (or square brackets [ ]). On the command lines, the braces are critical. It doesn't matter whether you have spaces between } and the next { or not. For reading commands from a file, braces are not needed if commands are on distinct lines, see Section I.7.

Commands are broken into two classes: normal commands and BUSIC language commands. All normal commands have a two-element minimal truncation. That is, all normal commands are distinguished already in the first two symbols, and any substring of the command that has at least two letters will work. For example, the basic command to get a report on the environment is

```
batutil {envrep}
```

The minimal truncation for that is EN. We'll often write ENvrep to emphasize this. Thus

```
batutil {en}
```

or

```
batutil {envr}
```

would work just as well as the full name. The basic commands are not case sensitive; so, {EN} or {En} or even {eNvRe} would do the same thing as {envrep}.

While to emphasize minimal truncation, we'll write ENvrep, please bear in mind that BATUTIL commands are NOT case sensitive although $ metastrings ARE.

The BUSIC language commands IF, WHILE, CALL, FOR, FORDIR, REPEAT, CASE, END, RET, ENDCASE, UNTIL, and BEGIN cannot be abbreviated although they are not case sensitive. To avoid accidents, the HACKER + command cannot be abbreviated but must be used in full.

In addition, much of the metastring translation acts as if it were commands. There is little difference between the HOur command which sets the errorlevel to the current hour and $H which returns the current hour in a string; therefore, {HO} and {ER $H} are equivalent.

Roughly fifty of the commands return to the user (i.e., you as a batch file writer) a number from 0 to 199. If the command in question appears inside {}, then that number is stored in the environmental variable RC (short for "return code"). Environmental variables are discussed at the start of Chapter V. If the command appears in [], then BATUTIL will exit without running the rest of the command line. It will also place this integer in the DOS errorlevel where you can test it with commands like

```
if errorlevel ....
```

(see Chapter VII). IF THE LAST COMMAND ON THE LINE RETURNS AN INTEGER AND IT IS PLACED IN {}, THEN the integer is both placed in the real environment and placed in the error level. For example, DRive reports the current drive with 0=A, 1=B, etc. If the current drive is D, then

```
batutil .... [DR]
```

will set the errorlevel to 3, while

```
batutil .....{DR} {echo hi there!}
```

would set an environmental variable RC to 3 and then echo `hi there!`. Thus, if you ran the SET command after BATUTIL, you'd find the line `RC=3` on your screen.

Every time a command would set the environmental variable RC, it also would set the internal variable $r accessible via metastring translation. If you wish to run BATUTIL in a mode where it doesn't use the user's environment, the {RC $} command will tell BATUTIL to only use $r and not set an environmental RC variable.

Some commands will take optional parameters which also appear inside the braces for that command. Thus, **@Disk** gives the free disk space. If no parameter is specified, the current default drive will be used. Otherwise, you can specify a drive as in

```
batutil {@D C}.
```

BATUTIL will object to incorrect syntax, so if you tried

```
batutil {@D 3}
```

then BATUTIL will exit (with an appropriate message if Verbose mode is on). You can separate the command from the first parameter by any of the following:

```
<space> <comma> = or :
```

Rules of separation of multiple parameters for those few commands which accept multiple parameters depend on the commands.

Some commands do explicit metastring translation of their parameters and others do not. For any non-BUSIC command, if you place the parameters inside **$!(...)**, then metastring translation will take place before the parameters are passed on. Thus, even though NUmberfiles does not normally do metastring translation, the command **{NU $!(...)}** would first do metastring translation on ... and then process the command

```
{NU —}
```

where --- is the metastring translation of ... This supermetastring translation even allows $Q, $N, and $F file input.

Before doing anything, BATUTIL scans the command line and makes sure that every command is a proper one for this version of BATUTIL. If not, it will exit. Otherwise, it will execute the commands one at a time. It does not check for parameter syntax until that command is reached. Thus, {@D 3} will halt BATUTIL when reached, but earlier commands will have already run.

You may place remarks in the BATUTIL command line by enclosing them inside {} with the REmark command as in

```
batutil {AT 4F}{CL}{REM clears the
screen in red!}
```

## I.6 Setting BATUTIL Options from the Environment

You can effect BATUTIL's behavior with five environmental variables: BU@, BU^, B4$, CUR, and BUSWAP. These are set with the DOS SET command or BATUTIL's SEt command.

When BATUTIL loads, before reading the rest of its command line, it looks for an environmental string of the form

```
BU@=string
```

and it first reads that string as if it were a command line and places the commands in it to be processed first. This is intended primarily to allow you to change the built-in default options to some other value. For example, to turn off beeps and change the default on GEtkey from flushing the keyboard to not, you'd use

```
BU@={NB}{NF}
```

This option is useful if you want to permanently turn on pause mode or European date mode. You'd use

```
BU@=P E {NB}
```

if you wanted to also turn beeps off.

Since BATUTIL looks in the environment, you place the information that you want there using the DOS SET command (or BATUTIL's SET command if you want!). For example, you might place a line

```
set BU@=P E {NB}
```

in your AUTOEXEC.BAT file. It is important to place NO spaces between BU@ and the =. Spaces after the = are unimportant.

There are four other environmental variables of interest. One is the variable B4$. At the end of the commands to edit your path and the environment ({PAthedit} and {EDitenv}) and after using help, a screen will appear reminding you of the fact that BATUTIL is shareware, limited to a 30-day trial period. You can suppress this screen by placing

```
B4$=I paid
```

in your environment. Obviously, having told you the secret, you can do it even if you haven't paid — let your conscience be your guide. The variable BU^ can be used to turn off ^Break handling; see Section I.11.

BATUTIL's RUn command swaps to hard disk or EMS. You can control where swapping takes place with the BUSWAP variable. This is discussed in Section II.7.

The final environmental string that BATUTIL pays attention to is CUR (for CURSOR). It looks for two possibilities:

```
CUR=on
```

Normally, BATUTIL restores the cursor when exiting, even if you have turned it off with the {CU –} command of Section III.7. If this string is present, the cursor is not turned back on, but its state is not changed.

```
CUR=off
```

When starting or exiting the program, BATUTIL will turn the cursor off if this string is found at that point. All other values of CUR are ignored.

## I.7 Reading Commands from a File

**Commands:** FCommand, INclude

Four commands are able to read information in from a file. The FCommand command, which is equivalent to the INclude command, reads new commands from a file. The ECho command displays text on the screen in colors you set before the echo command; PRetty displays text in colors that you can adjust as part of the string displayed; and MEnu displays a user-defined menu. The analogous file-reading commands are FEcho, FPretty, and FMenu. Each command takes two parameters: the first is a filename and is required; the second, a label name which is optional. If the filename is given without a drive or directory, then it is searched for first in the current directory. If not found by BATUTIL, BATUTIL will attempt to add an extension of BAT and look again. Then the same search is made in the directory that BATUTIL was loaded from (under DOS 3.0 and later) and finally in the DOS path. If still not found, BATUTIL exits and an error message is issued.

If no label name is given, then BATUTIL will start reading and processing the file from its beginning. If a label is given, BATUTIL reads the file from the beginning but searches until it finds a line beginning with

```
:label
```

(leading spaces before the : are ignored and labels are NOT case sensitive). For example

```
batutil {FE foobar.txt hi}
```

would look for a file named foobar.txt and then search for a line starting with

```
:hi
```

If the label is not found, then BATUTIL exits with an error message. Otherwise, the file is processed one line at a time until the next line beginning with a : is located. Lines whose first symbol is a ; are ignored (i.e., treated as remarks).

You can combine batch file labels and BATUTIL's labels to keep the extra lines to display in the batch file itself. For example, the following could be at the start of a batch file:

```
goto codestart :echostart
This line will be echoed
and this will appear on the next
   line
;but this line is a remark
The last line doesn't have an
   explicit CR added
:codestart
batutil {FEcho %0 echostart}
```

Because BATUTIL will add .BAT to a filename if it cannot find the file without that, the %0 will be properly interpreted (unless you happen to have a file with the same name as the batch file and no extension).

FEcho, FMenu, and FPretty are limited to reading in 25 or fewer lines. FCommand/INclude will read in up to 1,023 commands. You can nest INclude commands to get around the 1,023 command limitation, but BATUTIL is an interpreted language and such large projects should normally not be done using such a language.

You can place multiple commands on a single FCommand line by separating them with one or more of the four characters [ ] { }.

Please look closely at the demonstration batch files for an introduction of how to use the file commands effectively. A typical batch file that only calls BATUTIL commands including the RU command could have the form:

```
@echo off
batutil {INc %0 code}
quitbat
:code
BATUTIL commands
```

To make this work, you'd need a zero-byte batch file called QUITBAT.BAT in your path. You can make such a file at the DOS command line by typing

```
rem > quitbat.bat
```

FEcho and FPretty are discussed in Section III.7 and FMenu in Section IV.2.

BATUTIL stores the name of the last file you have asked to read commands, echo, pretty, or menu from and will reuse it if you use the filename %0 in one of these commands. See the sample batch files.

## I.8 BATUTIL's Line Editor

Four of BATUTIL's commands allow editing of line input: the EDenv and PAthedit commands, which allow you to edit any environmental variable; your path, the USername command which prompts for a string to be matched; and the $Q/$?/$N subcommands of BATUTIL's SEt, which place information in the environment. When such input is asked for, the following edit keys are active:

| | |
|---|---|
| <Enter> | accepts the current string and exits |
| <Esc> | restores the original value of the string; if that original value is empty, <Esc> blanks the line except, in that case, <Esc> with a blank line exits and returns an empty string |
| <Left>, <Right> | move the cursor by a character |
| <^Left>, <^Right> | move the cursor by a word |
| <Home>, <End> | to the beginning or end of the line |
| <Ins> | toggles insert mode which starts as overwrite mode; the cursor shape indicates what the current mode is |
| <Del> and <Bks> | do their usual single character functions |
| <^X> or <^Y> | blanks the entire line |
| <^End> | blanks to the end of the line |
| <^Home> | blanks from the start of the line to the current position |
| <^T> | blanks from the cursor position through the next blank space |

In addition, WordStar commands are accepted for most of these functions; for example, ^S is the same as <Left>, and the two letter ^QS (or ^Q^S) is the same as <Home>.

In situations where the quantity being edited has a prior value, that value is displayed when the line editor starts up and the cursor is at the end of line. Hitting an alphanumeric key (as opposed to a cursor key) as the first key will blank the original value which can be restored with <Esc>. The $N input must be an integer, and only integer input is accepted.

In each case, the input string has a maximum length which may be larger than the edit window on-screen. If it is larger, the line editor will scroll horizontally if the cursor moves to the start or end of the edit window.

The point is that the editor is quite intuitive and most users will have no trouble using it without explicit directions.

## I.9 BUSIC — A First Look

BATUTIL includes a full-featured language which goes way beyond the DOS batch language. We dub it *BUSIC* for **B**at**U**til's **S**tandard **I**nstructions and **C**ommands. Despite the name, it is closer to Pascal or C than BASIC. While it can be used at the command line, the use of {}'s gets tedious, and we recommend limiting BUSIC to INclude sets of commands. Chapter VI is devoted to this language, but we'll make a few general remarks here.

There are 32 internal variables. Two are special: $r is like the RC environmental variable, and $% tracks the last fileIO error when IOerrors are trapped. These two variables cannot be set with the set command but can be ECHOed with metastring translation or used in other places, for example, inside a comparison after an IF, where they are read.

You will most often use the internal variables $1,...,$0, ten in all. In addition, while they are more awkward to use, there are an additional 20 variables which are indicated by $ followed by one of the ASCII codes 130–149, that is $é, $â, ..., $ò. These can be set with commands like

```
SEt $1=....
```

The set is actually optional for these 30 variables, that is

```
$1=$Q
```

is the same as

```
SEt $1=$Q
```

and would place up an edit box whose value is stored in $1.

The commands:

```
++ $1
```

and

```
-- $1
```

(where $1 is any of the 30 internal variables that you can set) will check to see if $1 is number (if $1 is not, then there will be an error exit!) and if so increase or decrease it by one. If the variable is empty, it is treated as if it were 0 for purposes of these commands.

The keywords in the language cannot be abbreviated. We'll usually capitalize them for emphasis, but they are not case sensitive. After THEN or ELSE in an IF clause, after FOR, after a CASE option, and after a WHILE or REPEAT, you need to place a legitimate BATUTIL command, for example

```
IF $1>6 THEN ECho it is greater
  than 6.
```

In all of these cases, you can replace the command by the single keyword BEGIN and then place several commands each on a separate line (or separated by {}[]) with an END to indicate that you finished the set of commands started with a BEGIN. For example, to count the files in a directory starting with b, echoing their names to the screen, you could use

```
$1=0
FOR $2 in (b*.*) DO BEGIN
++ $1
ECho $1:$2$_
END
```

Our sample will indent commands between BEGIN/END pairs for readability, but leading spaces are ignored by BUSIC.

$R is a special metastring. It will read from a text file (see Section VI.5 for discussion of reading from files and writing to them). If the file you read from is **con**, then special handling is done and you read from the screen. You can read from absolute screen positions or positions relative to the current one.

BATUTIL's TRace command is discussed in Section VI.12. There is a powerful but dangerous option that allows you to directly read and write from/to memory and or a processor IO port. Because we'd not want you to do this by accident, you must turn this mode on within EACH running of BATUTIL with a {HACKER +} command. We discuss this in Section VI.13.

## I.10 String Manipulation, Arithmetic, and Date Arithmetic

BATUTIL includes special metastrings to handle string manipulation and arithmetic. $s (to be distinguished from $S=space) when translated manipulates a string. The syntax is

```
$s(X,string,Y)
```

where S is a single letter key and Y specifies parameters which are special to each value of X. For example,

```
$s(W,string,4)
```

would become the fourth word in the string and

```
$s(U,string)
```

would make the string Uppercase. Section VI.2 describes this metastring in full details.

$V (to be distinguished from $v=DOS version) does arithmetic. The syntax is to put an arithmetic string inside parentheses following $V. For example,

```
$V(1+4)
```

would translate to 5. Only integer arithmetic is done. Operations are summarized in Section VI.3. Multiple parentheses are handled, and there is metastring translation inside the (..) so that you can use $1, etc. and environmental variables. For example internally, BATUTIL translates ++$1 to

```
SEt $1=$V($1+1)
```

In Version 1.0, $E, $M, $D, $Y, $W referred to today in English, the month, day of month, year, and weekday. That remains true IF these are not followed by a (. If they are, BATUTIL will attempt to translate the argument into one of three formats:

1. Sees if it is number between 1 and 7305 and interprets it as a date between 1/1/80 and 12/31/99
2. Sees if it has the format MM-DD-YY or MM/DD/YY and interprets it in that format as an American style date
3. Looks for a filename with that name and interprets it as the date of that file

So, for example, if the file C:\dos\command.com has a date of 12/19/88, then all three of $E(12/19/88), $E(C:\dos\command.com), and $E(3276) would become December 19, 1988 after metastring translation. If the European date switch is turned on, then the date is interpreted as DD/MM/YY in place of MM/DD/YY.

The use of number of days after 1/1/80 wouldn't be useful if you had to count the days yourself. So, there is a metastring $J which counts that number of days. With no parameters, it is today and for example $J(12/19/88)=3276. Thus, for example, you could do date arithmetic with

```
batutil {EC the date 200 days from
today is $E($V(200+$J))}
```

## I.11 Control Break Handling

BATUTIL looks especially for the user to press ^Break. If this is pressed during the running of BATUTIL, then the message

```
Quit BATUTIL(Y/N)?
```

pops up. If you answer yes, then BATUTIL pops up a message:

```
Halt Batch file too(Y/N)?|
```

With either response to this question, BATUTIL is then halted. If you've also answered yes to the second question, then BATUTIL will turn DOS Break on and place a ^C buffer so the batch file should offer you the chance to terminate it. If you prefer running with Break off, you'll need to do that by hand if you use this emergency exit. This response is to ^Break only and NOT also to Ctrl-C.

If you answer N to the initial QUIT BATUTIL question, then BATUTIL will ask

```
Turn TRACE On(Y/N)?
```

Either way BATUTIL will continue, but if you answer yes to the TRACE ON question, then BATUTIL's single step debug mode is turned on, but not until the currently executing command is finished.

If you quit BATUTIL with ^Break but elect not to halt the batch file, BATUTIL exits with the errorlevel set to 255.

For third-party batch files, you may want to turn off ^Break to prevent the user from breaking at an inconvenient point. When BATUTIL loads, it checks to see if **BU^=no** is in the environment. If it is, then the ^–Break handler is not installed. So, just include set BU^=no in the batch file to avoid this premature exit.

Please see the BATUTIL.DOC file for Chapter 2 omitted here.

## Chapter III: Display Tools

***Note:*** Please see BATUTIL.DOC for sections III.1–III.9 omitted here. (Be sure to read about Bigecho in III.6; it's a fun command.)

## III.10 Sounds

**Command:** SOund, NSound

**Parameter:** SOund takes a single required number from 1 to 20 and an optional second number. NSound takes a -- or an optional +.

BATUTIL comes with ten brief sounds and ten tunes. The ten tunes were made with PIANOMAN. Each has a number.

The ten sounds are as follows:

1: ping
2: wolf whistle
3: random electronic sound
4: short buzz
5: tweet
6: alarm clock ring
7: buzzer
8: electronic sound 1
9: electronic sound 2
10: train with Doppler effect

The tunes are fragments from the following pieces:

11: Dance of the Clowns
12: Habana from Carmen
13: Sailor's Hornpipe
14: Mapleleaf Rag
15: Land of Hope and Glory (Pomp and Circumstance)
16: Porky Pig Theme ("That's All Folks")
17: Pop Goes the Weasel
18: William Tell Overture (Lone Ranger's Theme), Part I
19: William Tell Overture, Part II
20: Yellow Rose of Texas

You invoke a sound with BATUTIL by using the sound command which takes one or two parameters. The first parameter must be an integer from 1 to 20 and indicates the sound from the above list. The second parameter indicates the number of times to repeat the sound; if the second parameter is absent, the sound is issued once. For most sounds you won't want any repeats, but for sound 3 you'll want a repeat count of 15 or more, and sound 4 will do with a few repeats. The repeat count must lie between 1 and 60.

Thus

```
batutil {SO 3 30}
```

will repeat sound 3 thirty times. The William Tell Overture is broken into two parts, to allow you to take an action in the middle as in

```
batutil {EC CTRLALT Associates}{SO
18}{EC $Spresents}{SO 19}
```

If you don't want the SOund command issued, you can use the NSound command. This is intended primarily for use in the options set in the environment as in

```
set @BU={NS}
```

which would be useful if you normally had sounds in your batch files but were working late at night and didn't want to disturb others. The {NS –} command would turn sound back on even if there is a previous {NS}.

## Chapter VII: Tips and Examples

## VII.1 General Tips

After a BATUTIL menu, you will typically need to branch on a number of different possibilities. The commands

```
IF $r=6 GOto menu6
IF $r=5 GOto menu5
IF $r=4 GOto menu4
IF $r=3 GOto menu3
IF $r=2 GOto menu2
IF $r=1 GOto menu1
```

followed by choice 0 code will work, but the following is more elegant. If there are six menu choices, have labels menu0,..., menu6. Use

```
GOto menu$r
```

For an example of this, see the sample batch file BUDEMO.BAT.

It is most convenient to place the fmenu and fecho lines in the batch file itself. If the batch file is foobar.bat, you could use

```
{FEcho foobar.bat label}
```

but that will fail if the user renames the batch file, so use

```
{FEcho %0 label}
```

instead. To handle this possibility, BATUTIL looks first for the filename specified with a {fecho ....} command by name and if that fails it tries the extension BAT.

## VII.2 Returning to Your Initial Directory

You can arrange to return to your initial directory in a batch file, say one that changes to C:\123 and runs Lotus with

```
batutil {SEt dr=$n di=$p}
C:
cd \123
lotus
%dr%:
cd %di%
```

This uses DOS's environmental substitution.

You could arrange to save a set of your earlier directories and return to them with two batch files: PUSHDIR and POPDIR. PUSHDIR.BAT would read

```
batutil {SEt dr3=$x(dr2)
  di3=$x(di2) dr2=$x(dr1)
  di2=$x(di1)
dr1=$n di1=$p}
```

all on one line. POPDIR.BAT would read

```
%dr1%
cd %dr1%
batutil {SEt dr2=$x(dr3)
di2=$x(di3) dr1=$x(dr2) di1=$x(di2)
dr3= di3= }
```

In terms of directory manipulation, suppose you not only want to return to the original directory, but you have a program named silly in \foo which insists that its directory be in the path even if it is the default directory. You want to add it to the path but later remove it. Use

```
batutil {SEt dr=$n di=$p}{ad
C:\foo}
C:
cd \foo
silly
batutil {DE C:\foo}
%dr%:
cd %di%
```

## VII.3 Using BATUTIL in Your AUTOEXEC.BAT

A number of actions (special to an AUTOEXEC.BAT) make BATUTIL valuable. Your AUTOEXEC.BAT may take a long time to run because you run a defragmenting utility or use a FAT/root dir saver like DOS's mirror or you might copy a lot of files to a RAM disk. You might care to branch at the end of the batch file, say to run Software Carousel or not. Of course, you can ask a question, but what if you don't want to always stand around and would like to default to running Carousel if you aren't there? You could use (as a fragment of your AUTOEXEC.BAT)

```
batutil {EC Run Carousel (Y/N)?}{GE
   wa6 y n}
if errorlevel 2 goto nocar
carousel
:nocar
```

You could even have the time counted down by replacing the first line with

```
batutil {EC Run Carousel (Y/N)? $L
   secs left}{GE wa6 y n}
```

So, you've solved the problem of waiting around if you do want to run Carousel. But, what if you don't want to? With the above fragment, you'd need to wait around. Here is where the test of the lock keys comes in. Immediately before the first line of the above fragment, you could put

```
batutil {QL C-}

if errorlevel 1 goto nocar
```

Then, if you hit Capslock before you go off, Carousel wouldn't run. Actually, you could avoid an extra running of BATUTIL and two lines of the file by using:

```
batutil {QL C-}{IF $r=1 THEN GOto
   halt2}

{EC Run Carousel (Y/N)?}{GE wa6 y
   n}
```

as the first line of the fragment. For then if Caps was set, the GOto halt2 command causes BATUTIL to exit with an errorlevel of 2 just as if you answered no!

Another problem that you'll often want to cope with in a batch file is some operation like defragmentation that you only want to run once a day. Here is a fragment to do that:

```
batutil {TO newday.tst}
if not errorlevel 1 goto already
echo a >newday.tst
STUFF TO DO ONCE
:already
```

What this does is, if the errorlevel is exactly 0, then one skips to the label `already`. Otherwise, the file newday.tst is given today's date so that the next time it is run, **batutil {to newday.tst}** will return errorlevel 0. But what if you sometimes work past midnight and don't want the newday stuff done until morning? Replace

```
batutil {TO newday.tst}
```

with

```
batutil {TO 5 newday.tst}
```

which will make the changeover at 5AM rather than midnight.

## VII.4 A Clean Sweep

Here is a simple BATUTIL code fragment that will delete all *.BAK files anywhere on disk.

```
SEt buswap=!
FORDIR $1 IN () DO BEGIN
$2=$A($1\)
NU $!($2*.bak)
IF $r > 0 THEN BEGIN
RUn command /cdel $2*.bak
ECho $1:$r files erased
END
END
```

Note the use of $2=$A($1\) to place a single trailing backslash as explained in Section VI.10.

Note also that the supermetastring **$!** in the **nu** command is necessary because metastring translation is not done on that command.

## VII.5 Directory Lister with Smart Wildcards

You may want to know how to get a name listing of all files whose names includes a string anywhere in the filename; see, for example v10, n16, pg 424. Here's a way to do it with BUSIC:

```
@echo off
if '%1'=='' goto error
batutil {$1=%1}{IN %0 code}
for %%a in (%foo##%) do echo %%a
set foo##=
goto end
:code
$2=$s(S,$1) $5=$1*.* $4=?
FOR $3=8 TO 2 DO BEGIN
IF $2=$3 THEN GO next
$5=$5$S$4$1*.*
$4=?$4
END
LA next
CASE $2 OF
1: $5=$5$S*.$1*$S*.?$1*$S*.??$1
2: $5=$5$S*.$1*$S*.?$1
3: $5=$5$S*.$1
ENDCASE
SET foo##=$5
:error
echo a parameter is required
:end
```

## VII.6 A Two-Year Old's Delight

Here is a simple batch file which delighted Barry's two-year-old daughter:

```
for %%a in (1 2 3 4 5 6 7 8 9) do
   batutil

{GE IN}{SO 1%%a}
```

This will wait for the insert key and then play 9 tunes. The INsert key was especially picked for two reasons: it is large and easy for little hands to reach. It is not a repeating key so that it is less likely that several keystrokes will get through and abort the tunes.

## VII.7 Sample Batch Files

You will learn a lot about how to use BATUTIL by studying the sample batch files provided. We'll make some comments about them here.

The complicated contortions in the BATUTIL 1.0 manual are no longer needed because we can use internal variables on the whole. Notice that we use the **{RC $}** command to prevent RC from being placed in the environment.

BUDEMO {FEcho %0} and a variant on the goto fmenu$r tricks are discussed in Section 1 (we use CALL in place of GOto). It turns the cursor off with the

```
set cur=off
```

command.

Most of the separate batch demos are called with a simple use of FCommand. Notice that this use allows the same code to be used to call the subsidiary batch files from within BUDEMO.BAT as well as stand alone.

Notice how BUDEMO.BAT sees if BATUTIL.HLP is in the current directory or on the path and stores the information in $Å. It then calls the appropriate menu with a

```
FM $!(%0 fmain$Å)
```

The supermetastring **$!(...)** is needed because FM does not normally do metastring translation.

Because calling help requires a separate running of BATUTIL, we need some fancy footwork with DOS batch files to call help from the ninth menu choice.

SOUNDMEMO.BAT is also straightforward. Note the way the fpretty command is used to produce the second screen with its special box.

SHOWDEMO.BAT's most interesting feature may be the last panel that illustrates how to use the fpretty command for a display of different columns in different colors. Note the use of the $$.

Note INPUTDEM.BAT's use of **$L** with spaces after it.

Note that EQUIP.BAT gets all its information and saves it in internal variables and then displays it all at once.

The study of these samples should give you an idea of how to use BATUTIL.

# CMFiler

**Version 5.35**

**Copyright © 1991–92 Charles F. Martin, NoVaSoft**

## Introduction

CMFiler provides a quick, simple, one-touch operating environment for your IBM-compatible 80x86-based computer running under DOS Version 2 or later. While its name implies it is a file manager, it is much, much more. Once you have used CMFiler for a few sessions, you will not want to struggle with the DOS COMMAND.COM command processor or your other file managers ever again! CMFiler was painstakingly written in Assembly language over a three-year period, carefully factoring in the suggestions of a dozen of the world's pickiest software critics. It is compact and lightning fast.

***Note:*** See the file CMFILER.DOC on diskette for complete documentation. The Overview section and Chapter 1 of the tutorial are printed here along with a command summary and some information on the CMFiler editor (CMeditor).

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the CMFILER files in the directory \SECRETS\CMFILER. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the CMFILER files to a directory that is included in the search path.

**File Contents**

There are ten files in the CMFiler package.

| | |
|---|---|
| CMFILER.COM | The kernel of the CMFiler program |
| CMFILER.OVY | Its supporting program overlay |
| CMFILER.DOC | The full documentation |
| CMFILER.BRF | A brief summary of features for quick reference |
| CMFILER.REG | A file containing the registration form |
| DESC2NAR.COM | A program for 4DOS/NDOS users for converting the DESCRIPT.ION files containing file and directory notes into the CMFiler NARATIVE.CF file format |
| DESC2NAR.DOC | The documentation for DESC2NAR.COM. DESC2NAR is provided as a freeware adjunct to the CMFiler shareware disk/file manager for 4DOS/NDOS users |
| CHANGES | A brief compilation of the changes and bug fixes since Version 5.22 |
| READ.ME | A file with installation instructions |
| FILE_ID.DIZ | A description file used by BBS system operators |

CMFiler is capable of doing everything your DOS COMMAND.COM processor or other file manager can do and more but with EASE! — displaying two directories side-by-side, copying one or more files from one directory to another, backing up files in the same directory, deleting and renaming files and directories, setting file and directory attributes, concatenating (stringing together) multiple files, viewing and editing files, making and removing subdirectories, printing files in the background while you perform other tasks, setting disk volume labels, printing directory listings, keeping notes, displaying two disk structures side-by-side as "trees," and performing a full set of operations on their structures and executing child processes — all with a minimum number of keystrokes and maximum "transparency."

Operations are keyed to single-letter mnemonics, making CMFiler intuitive and easy to learn. For example, the letter C means *copy*, the letter D *delete*, and so forth. Almost everything you need to know about CMFiler is contained in a two-line prompt at the bottom of the screen or can be summoned instantly with a single keystroke (surprise! H for "help"), with no delay for external file reading, since the help is embedded in the program code. There are only a few things for which you will need to refer to this manual once you begin using CMFiler.

CMFiler employs the BIOS (Basic Input/Output System) and DOS disk service routines which are embedded in ROM or loaded into RAM when your computer boots, and the DOS file operation safeguards and error flags are employed, such as denying access to remove non-void subdirectories. I have added a few of my own, such as protection of files against being overwritten by zero-length files of the same name (resulting from an occasional DOS read-write failure).

CMFiler performs most operations immediately and does not ask for confirmation as other file managers do, unless file destruction is involved. You must confirm with a separate keystroke, for example, that you really *do* want to delete files, or that you really *do* want to overwrite a more recent version of a file during a copy operation. Otherwise, CMFiler takes for granted that you are doing exactly what you had in mind on the basis that a mistake can be recovered from with just a few more keystrokes. The mass operations, like copying large numbers of tagged files, all have an Esc bailout.

## Program Logical and Physical Organization

The CMFiler program code is organized logically (in the programmer's sense) into three sections or modules. These will be referred to variously as the *main module* or *main screen*, the *editor module* or *editor*, and the *tree module* or *tree functions*. Each module performs some major function or group of functions which is fundamentally different from the rest, discussed further below. On the other hand, the program is broken physically into two main pieces; in this case two files, namely the file CMFILER.COM, sometimes referred to herein as the *kernel*, and the file CMFILER.OVY or the *overlay*. The kernel contains the most vital part of the main module, the starter, and the overlay, which the kernel loads contiguous to itself in memory, contains the rest of the main module, plus all of the editor and tree modules. A memory map is included in this manual for those interested.

The main module displays one directory on the left side of the screen or two directories in side-by-side panels. The subdirectories of each directory are always shown at the top of the listing in alphabetical order followed by the files in one of seven user-specified ordering schemes. Commands operate on a subdirectory or one or more files in one panel (the source panel). Some commands, like Copy, write to the directory in the other panel (the target panel). The main module gives a somewhat myopic or microscopic view of your disk, one directory at a time, but does so with the most complete set of file and directory services available in the file manager. The other two modules (editor and tree) are accessed from the main, and the editor may also be accessed from the tree.

The editor is called to create, edit, or view a file. To view or edit an existing file in the main module, just put the cursor on it and press E or Enter. To create a file, press F and type in a name for the new file.

The tree module is called from the main module with Shift-T. It displays the entire directory tree structure of one disk in the left panel, or two trees for the same or different disks in side-by-side panels. Whole structures (branches) of a tree — or if you wish just the files in one directory — may be copied, moved, or deleted in their entirety. Some file manager documentation calls this kind of capability *prune-and-graft*. This capability is macroscopic in scope compared to the main modules as it deals with the big picture of your disk organization. But CMFiler does much more than most file managers. In addition to the macroscopic prune-and-graft-type operators, you may do some microscopic things as well, like display the files in each subdirectory as you navigate through the tree, find a specific file anywhere on the disk, search for a text string in all or some of the files on the disk — even view or edit a file, and delete it without having to return to the main file services module. This latter feature, in particular, makes disk cleanup a breeze!

## Security Features

CMFiler incorporates security features which support its use on systems containing sensitive information.

- You may specify a password and change it at any time during a session which must be given back to reenter CMFiler once it has gone to screen-saver mode (about 1½ minutes after last keystroke from main screen display). This password is the bottommost field in the data entry screen presented by the key combination Shift-F10 (discussed in further detail in Chapter 4).
- You may force CMFiler into screen-saver mode with the key combination Ctrl-S. This is so that if you want to secure access to your computer immediately you don't have to wait the 1½ minutes for the screen saver and password to automatically be invoked.
- If you set the option **Clear I/O buffers if located?** in the Shift-O Options screen, CMFiler will attempt to locate and overwrite all the DOS input/output buffers with the continuous string `CFCFCFCF...` before starting any file writing operations. This operation ensures that the file being written does not contain any stray data in the "slack space" of the last sector written that was hanging around from a previous I/O operation. ( ***DR DOS USERS:*** The option to clear I/O buffers should NOT be invoked if you are operating under DR DOS. This operating system responds very ungracefully to the buffer clearing feature. For this reason, I have set the default option to NOT clear I/O buffers.)

CMFiler also employs a rudimentary checksumming routine to check the .COM and .OVY files each time they are loaded. This routine will sense any changes in the execution code and fixed data areas of either file and warn the user of the corruption. Files may become corrupted as a result of an operating system error during data transmission, a sector going bad on a disk, or external tampering by a person or a virus. This check-summing feature can detect (and has detected) the presence of some older file infecting viruses but may be foiled by modern stealth viruses. DO NOT count on it for virus protection. There is no substitute for good antivirus software. I use and strongly recommend Wolfgang Stiller's Integrity Master. He is President of Stiller Research and a member of the ASP's Virus Information Panel, and a recognized expert in the anti-virus community.

## Monitor Options

CMFiler should work with all reasonably current IBM-compatible CGA/EGA/VGA and monochrome monitors. Although, I have heard of some problems with machines under very early BIOS versions.

You may specify a type of monitor one of two ways:

- When first run, if CMFiler cannot find a .CFG file containing previously specified user option data and does not sense a monochrome monitor installed (which only supports one color set), it knows it cannot tell for sure what you have installed; then it will ask you to specify a monitor type by pressing C, L, T, or M. Your choice is immediately recorded by CMFiler, creating a .CFG file. If you have an LCD monitor other than Tandy, you may invoke a generic LCD color set with the letter L. For Tandy LCDs, use T. (Tandy, for some unfathomable reason, inverts the function of the high-intensity bit in the color attribute.) If you have a monochrome monitor being driven by a color card, CMFiler senses the color card's presence, not the monitor's. Press M for this case to force the monochrome color set.
- You may also put one of these four letters in the command line, preceded by the character @, and bypass the initial question.

## Precautions and Limitations

There are a few precautions and limitations to be aware of:

- Terminate-and-Stay-Resident programs should NOT be launched from CMFiler. At best you will end up with a fragmented memory when you exit CMFiler, and at worst you will have a SYSTEM CRASH.
- CMFiler requires a minimum of 197K of free memory to run. This permits generous memory allocations for the directory listings, a print spooler buffer, and a large copy buffer. During application launching, however, the resident portion of CMFiler can be made to occupy as little as 19K using the Small footprint option of the Kernel command (letter K pressed from the main screen — discussed more in Chapters 1 and 4).
- Maximum directory size recognized by the main module varies from 300 to 1,200 entries, depending on free memory available. If the directory size limit is reached, an informational note is given. The rest of the directory is inaccessible, as though it were hidden. This is a benign, non-damaging limitation.
- The algorithms used by the editor module place certain limitations on it in edit mode which do not apply in view mode. If you attempt to edit a file which exceeds these limitations, the editor reverts to view mode automatically:

  Files too large to fit in available memory in one shot.

  Files with more than 16,380 lines.

  Files with lines longer than 8,190 characters if editing in NoWrap mode.
- The tree module limits the number of entries in a directory tree structure to 505 total subdirectories and the total files to 3,488.
- The main and tree modules both limit the depth of directory nesting to eight levels (e.g., **C:\1\2\3\4\5\6\7\8** is an example of the most deeply nested path allowed) and the length of path specifications to 66 characters (this is a DOS limitation).

- A limitation of DOS itself, which users frequently run into and are puzzled by because of the cryptic error message DOS returns, is the maximum number of entries permitted in the root directory. This number is 112 for 5.25" disks formatted at 360K and 3.5" @ 720K, and 224 for 5.25" @ 1.2MB and 3.5" @ 1.44MB. The error message returned is `Access denied creating file`. Be aware that the volume label and each subdirectory is an "entry," as well as each file.
- Finally, CMFiler does not yet support mouse or 43/50 line EGA/VGA display capability. If there is overwhelming request for addition of mouse support or 43/50 line display, I will consider adding it.

## Additional Note for Windows Users

Though not designed as a Windows application, CMFiler appears to be operating satisfactorily as a DOS program in the Windows environment. One nuisance feature will probably need to be corrected if you are running it under Windows, however. The default assumption on return from screen-saver mode is that the disk directories have not been altered while CMFiler has been idling. Under Windows, this is not necessarily a valid assumption since another foreground application may have operated on the directories CMFiler was selected to when it went into the background. Therefore, for Windows users there is a switch which may be set using the command Shift-O (where O is for Options) telling CMFiler to refresh the directory listings on return from the screen saver.

## Tutorial — Chapter 1: File Handling

### Getting Started, or the kernel and the overlay

As mentioned above, I refer to the 19K CMFILER.COM file as the *kernel* or the vital part of the main module and the 82K CMFILER.OVY file as the *overlay*, which contains the rest of the main module and the editor and tree modules. Physically breaking the program into two files served two purposes — it got around an arcane 64K limit on .COM files imposed by DOS, and it let me give you two options for how much of the program to leave resident in memory when launching applications from CMFiler (more in Chapter 4).

For now, copy CMFILER.COM and CMFILER.OVY into the root directory of your hard disk or a working floppy using your good ol' DOS COMMAND.COM processor and save a write-protected copy for your library. With the DOS system prompt showing the drive these two files are now on, run the kernel CMFILER.COM by entering cmfiler or CMFILER. The DOS command processor will load and execute CMFILER.COM, which in turn will find and load the overlay CMFILER.OVY. The whole 101K (19+82) is now residing in memory ready to respond to commands.

The first thing you will likely see is a request for monitor information. CMFiler cannot distinguish between a color monitor and an LCD monitor, and the default color set for the color monitor is a poor choice for LCD monitors. When you see the message press C, M, L, or T (some Tandy LCDs employ a backward convention on the high-intensity attribute, and thus a separate color set has been dedicated to them). As soon as you press one of these letters, you will see disk activity. CMFiler is creating a file of user configuration data (called CMFILER.CFG) in the same directory the .COM file is in. It will update this file as you select other setup options.

### Rename CMFILER to spare my fingers

Now, feel free to rename CMFILER.COM to CF.COM for short and the overlay and configuration files from CMFILER.OVY and CMFILER.CFG to CF.OVY and CF.CFG. (It is important to keep all of these files named THE SAME, by the way.) In fact I will refer to them simply as CF from here on, since that results in fewer keystrokes for me! Put the cursor on the line reading CMFILER COM and press the letter R (for *rename*).

A window will open up at the bottom that says:

```
Rename CMFILER.COM
    to >CMFILER.COM<
```

with the cursor blinking on the first position in the data entry field denoted by the > < pair. (CMFiler is assuming that you want to rename just the CMFILER.COM file since the most frequent use of the rename facility is renaming single files. CMFiler is offering you the same name as the default entry in the new name field since often when you rename a file you may be only changing a few characters.)

In this case, we want to rename all the files that start with CMFILER; that is, we would really like to use the DOS wildcard convention *. If we were renaming this family of files from the DOS prompt, we would type in the command **ren cmfiler.* cf.***. In the CMFiler rename facility, the upper line is the first parameter of the DOS rename command and the lower line the second. We therefore want the upper line to read CMFILER.*. To do this press either the up arrow or PgUp, and press Tab to move over to the C in COM. Type * and press Enter or down arrow. Now the window reads

```
Rename CMFILER.*
to >CMFILER.* <
```

Simply type **CF.*** and press Enter. The files are now renamed.

This quick exercise sounds complicated when you read it, but won't be when you try it. And it illustrates a few nice touches I have tried to build in to minimize your keystrokes.

From here on the explanations are not so detailed. I recommend you fiddle with scratch disks until you are familiar with all of CF's capabilities. If you are running DOS Version 2, put CF, both the .COM and .OVY files, on one of the paths defined in a DOS path command so that it can find itself after running other applications. This is not necessary under DOS Version 3 and later.

### The screen

You should by now have noted an economy of screen usage. The left panel shows the contents of the root directory on the default drive. Line 1 of this active panel shows the disk volume name if any and a prompt area for four display enhancement features: the key combination Ctrl-O (denoted by ^O to save space) allows selection of one of seven file Ordering schemes, ^H toggles the Hide mode switch, ^C toggles the Compare mode switch, and ^M allows specifying a file Mask. Line 2 shows the path to the current directory, whose first 20 entries are displayed in lines 3–22. Line 23 gives vital information about the disk, such as room left, and lines 24–25 contain an abbreviated help screen.

The thick versus thin sections of the vertical line to the left of the file list denote the relative position and size of the current screen display within the full directory listing.

If you do not press a key within about 1½ minutes, the screen goes into screen-saver mode. Just press any key to return to the main display. You may force the screen-Saver mode with Ctrl-S.

### The help area

The two-line help area at the bottom of the screen contains a set of abbreviated cues to assist with the recollection of the mnemonic single-key commands. With no modifier keys pressed [i.e., the Shift-, Alt-, and Ctrl- keys all up], the help area shows the operations available with the unmodified keys with the mnemonic code highlighted. These are typically the most frequently used operations; the letter C for *Copy file(s)*, E for *Edit file*, D for *Delete file or directory*, T for *Tag file*, the number 1 for *toggle the file Read-only attribute*, and so forth.

Additional commands are available which use similar mnemonic devices, but with the keyboard modified by Shift-, Alt-, or Ctrl-. Press one of the Shift- keys, and notice the change in the bottom lines. These are the commands enabled by the letters shown in highlight and modified by the key you are holding down. Press Ctrl- and Alt- in turn. Just be aware that the visual cues for the modifier keys are there when you want them.

### More on the modifier keys

Incidentally, there is some rationale for which of the Shift-, Alt-, or Ctrl- keys was used as the modifier for a particular modified operation, and there are some devices which may help you remember the modified keys as well as the more straightforward mnemonic devices of the unmodified keys:

- The modifier Shift- is often used for operations involving transfers of control or data to or from the other panel. For example, pressing Shift-Enter when the cursor is on a subdirectory entry causes this subdirectory in the other panel to be displayed and moves you across to it. Other commands which follow this convention and which you will encounter in more detail are Shift-Left/Rt Arrow, Shift-P, Shift-*, Shift-\, and Shift-F.

- The modifier Ctrl- is most often used to toggle the state of a switch or tags — that is, to invert something. For example, Ctrl-H toggles the state of the hide switch , i.e., if Y (yes or on) it goes to N (no or off), if N it goes to Y. Other examples include Ctrl-O, Ctrl-M, Ctrl-C, Ctrl-A, and Ctrl-L.
- The modifier Alt- is most often used just to indicate an operation associated with a mnemonic alphabet key, but for which the unmodified key was already in use as a mnemonic for a more frequent operation. For example, C means *Copy* and is one of the most frequent operations; Alt-C means *add file spec to Command tail* and is used far less frequently. (In the editor, Alt-modified keys are used a LOT, since the unmodified keys are used for typing text!)

### Two panels

Each of two panels corresponds to a path to files in the DOS vernacular. Open the right panel by pressing the right arrow. You will be prompted at the bottom of the screen to specify a drive letter. (Lines 24–25 become the dialog area for information to you about what CF is doing or what it is waiting for you to do.) Just press the letter corresponding to a valid drive; don't press Enter, just a letter key. The root directory of that drive will be shown in the right panel in the same format as the left panel. The bright yellow color of the path specification on line 2 and the blinking cursor now identify the right panel as the source path. The left panel has become the target or destination path for copy and append operations.

Whenever CF is in compare mode and the cursor is on the name of a file in the source panel that also happens to exist anywhere in the target path, the target panel display is adjusted so that the file appears in the panel, and its date/time signature is put in high-intensity to catch your eye and show you the duplication. If the date, time, and size of the two files are not the same, the date/time signature of the NEWER file will blink. You can toggle compare mode off and on with the key combination Ctrl-C (the mnemonic is compare).

### Color palette

For display in CGA/EGA/VGA systems, CF has four choices of color palettes. The command Ctrl-P (for Palette) lets you cycle through the choices with the spacebar and select a different color scheme with Enter or return to the original one with Esc. The choice is permanently recorded in the .CFG file (discussed in more detail later).

### Alphabetic case options

CF also has four options for the alphabetic cases used in the panel displays. The command Ctrl-E (for casE) lets you cycle through the choices with the spacebar as in the Palette above which are 1) files and directories all uppercase, 2) directories in uppercase, file in lower, 3) all in lowercase, and 4) all in modified-Tauck convention where all letters are lowercase except the first and any that follows a non-alphabetic character (e.g., Cmfiler.Com or Read-Me.1St). I have found this last convention by far the easiest to read.

### Cursor movement

Shift back and forth with the left and right arrow keys. You are in effect switching source and target paths. Use the up/down arrows, PgUp/PgDn, Home, and End keys to move the cursor within a panel. PgUp moves the cursor to the top of the panel if it was not already there, and then moves up 19 lines in the directory. PgDn is similar for the opposite direction. Home goes to the first line of the directory; End, to the last.

The command G (for *Go to...*) lets you type in a file name to move the cursor to within the directory listing. As you type, the cursor is repositioned to the first file described by the character string you are building. When you have come to the file you want, press Enter or Esc. Alt-G is the equivalent Go to. . . for subdirectories.

In Compare mode whenever there is a duplicate filename highlighted in the opposite panel, you may jump directly across to it with the move Shift-Lf/Rt Arr, vice the unmodified arrows.

### Switching drives or disks

CF cannot tell when you have changed disks in the drive whose contents are shown on one of the screen panels. You have to tell it by putting the cursor in that panel and pressing Shift-R (for *Refresh*) or N (for *New drive*) and the letter designator for that drive at the ensuing prompt. If you want to switch drives, say from A to B for the right-hand panel, put the cursor to the right and press N and B.

Sometimes when you are doing single-panel operations like constructing a note set or editing files, it is a distraction to have both screen panels open at once. To close the right-hand panel and return to single left-hand panel display, put the cursor in the right-hand panel and press N and Enter.

### Subdirectories

Make a new subdirectory in the source path by pressing M (for the DOS command *Mkdir*) and entering a name. Find it in the display in alphabetical order, move the cursor to it, and press Enter. Note the new path on line 2 of the screen display and only the <Parent> entry in the file listing. Go back to the parent directory by pressing Enter with the cursor on the <Parent> entry or P with the cursor anywhere in the panel. You can navigate down and back up through the directory levels in a path this way, one level at a time. If you are several levels deep, there is another quicker way back to the root directory than by hitting P repeatedly, and that is the backslash key \. It works like the DOS command cd \.

To put a subdirectory from the current source panel list into the target panel, put the cursor on its name and press Shift-Enter. The subdirectory is listed in the opposite panel, and the cursor shifts over to it. In similar fashion, Shift-P puts the source panel's parent directory into the opposite panel, and Shift-\ puts the source's root into the opposite panel. Finally, Shift-* puts the source directory itself into the target. This is handy for quickly setting up the same panels side-by-side for ZIPping/unZIPping, discussed in Chapter 4.

Remove a subdirectory by placing the cursor on it and pressing D (for *Delete*, which I use interchangeably for file deletion and subdirectory removal). Note that the subdirectory has to be empty first — a DOS safeguard.

### Copy some files, then do it with tags

Put the cursor on a filename and press the letter C. This copies the file from the source to the target path. Tag several files with the letter T or the spacebar. Now press C to copy this group from source to target. Clear all the tags with A (tag/untag All). Press A again, and see that all the files are now tagged. Untag an individual file with T. (T actually toggles the state of an internal tag bit assigned to each file and used for temporary marking purposes only. A clears all the tags if any were set or sets all the tags if all were clear. No information is changed on the disk itself. These are volatile tags maintained only until the directory is reread for some reason, such as a file deletion or a copy operation into the directory.)

CF looks first at the space available on the target path before it starts to copy. If it doesn't see enough room free, it doesn't start the operation and alerts you to this limitation. This prevents write errors, messed up file allocation tables, and incomplete files that can result when space runs out during a copy operation.

CF has several special features in the copy operation:

- If an identical file exists in the target path — same name, extension, date, time, AND size — CF does not normally copy the source to the target as this would be wasted motion on the assumption that the files are identical. This no overcopy feature is controlled by one of seven switches which may be toggled in the user setup menu brought up with the command Shift-O (for user Options). It is sometimes useful to change its state.
- If a file by the same name but newer date/time is found on the target, CF will ask you specifically to verify that you really do want the newer file overwritten.
- If the source file has length 0 bytes or if the source and target files have the same date/time but the source is smaller, CF will ask for overwrite confirmation. This provides some protection against overwriting a good file with one which has become corrupted by a previous copy error.

- If a file by the same name but with the read-only attribute set is found on the target, CF will ask you to verify that you want it overwritten. (Likewise, if the target file is hidden and the hide switch is on.)

### Tagging, backing up

With a couple of files tagged, look at the line just below the last line of the directory listing. You will see the space in use in the current directory (KB Used), the amount of disk space occupied by any tagged files (Tagd), and the space still free (Free). Also, the current number of files tagged in the line below will appear after each tagging operation. Disk space is expressed in kilobytes, in integer multiples of the disk media cluster size, so what you see is the amount of space ACTUALLY tied up on the disk. (If your disk is a 5.25" DSDD floppy, its clusters are probably one K each. Every file takes up at least one cluster so a 1-byte file takes up one K of disk space, the same as a 1,024-byte file.)

Assuming the amount of space represented by tagged files is less than or equal to space available, press B to back up all the tagged files. CF's convention for assigning backup filenames in this operation is to reuse the first two letters of the extension (filling blanks with exclamation points), and then make the last letter of the extension a tilde character (~). Thus, the backup should always immediately follow the primary file in any alphabetical listing.

### Moving files

With the panels selected to different subdirectories on the same disk, one or more files may be moved from one subdirectory to the other. This operation does not read and write the file data clusters but only changes the subdirectory table entries so large files may be moved around quickly. Tag files if desired as with Copy and press the move command Shift-C (instead of C, think of moving as just another kind of copying, but you are shifting files to a different directory instead).

If the conditions are not satisfied for moving the files (e.g., the directories are not on the same disk), CF will copy the files to the target path but will not delete the tagged files in the source. You may delete the tagged files with two more keystrokes if you wish, but CMFiler does not assume this is what you meant. This encumbrance is deliberate. Because CMFiler is so fast, I have made it require confirmation whenever any file destruction is requested.

### Deleting a file

With the cursor on a file and no files tagged, pressing D results in deleting the file under the cursor after confirmation. If one or more files are tagged, they will be deleted after confirmation, not the file under the cursor. Alt-Tags discussed below are treated the same as normal tags as far as the delete operation is concerned. On hard disks, the default configuration of CF actually performs what I will call a soft delete for this operation. The files are not deleted using the DOS delete function but rather are redated to the current date and time, and moved into a directory created by CF called CF_TRASH.CAN. If you delete files from your hard disk by mistake, they are RELIABLY recoverable just by switching to the CF_TRASH.CAN directory and moving them back to the directories they came from using Shift-C.

If you delete a second file by the same name as a file already collected in the CF_TRASH.CAN directory by a previous soft delete operation, CF tries to rename this second file by replacing the last character in the extension with a 1. If this name is already in use, it tries to rename with a 2 instead, then a 3, and so on through 9. Thus, you are assured under all reasonable circumstances of not losing any deleted files, even duplicates. The only problem is that you may have trouble telling which duplicate is the original, as CF redates each of them during the soft delete.

Each time you select a new drive in the main module, CF looks to see if it is a hard drive (A and B are always taken to be floppies), creates the directory CF_TRASH.CAN if it is not already present, and then looks in CF_TRASH.CAN for the presence of files with any date earlier than the current date. If it finds any, it will ask you if you want to purge them — that is, perform the hard DOS file delete on yesterday's trash. You have the options **yes**, **no**, or **Enter to view** to see what is in the trash can. You probably want to keep the CF_TRASH.CAN as uncluttered as possible just to avoid tying up disk space needlessly. If you prefer to "empty the trash" less often than daily, be prepared to be pestered with the same question the first time you select that disk each day! (The D command always performs a hard delete in the CF_TRASH.CAN directory; this is the one exception.)

This soft delete facility is NOT intended as a means of backing up files, but rather is built in SOLELY for the purpose of RELIABLE recovery from inadvertent file deletion.

An additional delete option is available, Ctrl-D or hard delete. This operation performs the DOS delete, always regardless of the type disk selected. Use it when you KNOW you will not want to recover the deleted files.

And if you are REALLY sure of yourself, you can turn D from soft delete to hard delete. There is a switch available for this purpose in the Shift-O user Options menu mentioned above.

### The tagging convention expounded on

For all the above commands (copy, delete, move, back up), the operation is performed on ALL THE TAGGED FILES in the source panel if ANY are tagged and only on the file at the cursor if NONE are tagged. Some file managers offer different commands for copy tagged and copy file at cursor, and so forth. I have always found this unnecessarily complicated. For the FEW occasions in which you have a bunch of files tagged to do one of these operations (say you want to copy them), and you discover just before you start that you really wanted to do some other operation on just one of them first (say you realize one of them is out-of-date and you want to delete it), you will have to either untag them all with A, do the operation on the one file and then retag and do the original operation, OR just postpone the one-file operation. For this example, it is easier just to toggle the tag on the one out-of-date file off with T, copy the other bunch, clear the tags, and delete the one. It becomes just a matter of a little thought about the order in which you do things.

The view operation (discussed in Chapter 2) does NOT clear existing tags. So, suppose you are cleaning up a disk, tagging files that you recognize by name as no longer needed in preparation for a single, massive delete, and you come to one you aren't sure about. Put the cursor on it, press Enter (the view command), browse through it and decide if it's a keeper, and Esc from view mode. Note the previously placed tags are still there, and the cursor is still on the mystery file waiting for you to decide whether or not to tag it.

### Renaming a file OR a directory

Since we've already done this, I'll just briefly say that it works for directories, too. A word about the wild card character *: In the earlier exercise, we used it in the first window to declare that we wanted to rename all the files with CMFILER as the name, and any extension, to CF with the same extension. You may use the * in the name field of the first window instead of the extension but not in both. Used in the name field, it means change this extension, wherever it appears, to this other extension. This option is less useful.

If used in the first data window, the * MUST also be used in the second window in the same position. CF always senses the use of * in the first window and seeds the second window with this character in the right position.

The * may be used in the second window, even if not in the first, as a shorthand for *keep the same name (extension)*. For example, if you wanted to rename just CMFILER.DOC to CF.DOC, put the cursor on CMFILER DOC in the source panel, press R, type **CF.*** in the lower window, and Enter. Saves some keystrokes. This is a wildcard convention taken from DOS. (Sorry to you DOS heavies: For simplicity, I chose not to complicate things by including ? in the wildcard library for this facility!)

### Some additional tagging options

In addition to T (which toggles the state of an individual file's normal Tag) and A (which clears or sets All tags), there are some more tagging operations. Alt-T applies an *append tag* discussed later. Alt-M tags all the files in the panel with the same naMe as the file under the cursor; Alt-E, same Extension. Alt-D tags all files in the panel with the same Date as the file at the cursor, Alt-N tags all files Newer, and Alt-O, Older. Alt-P tags in the oPposite panel all the files with the same names as files tagged in the source panel. (This is good for updating to a floppy only the files which were already on it.) And finally, Ctrl-A toggles the state of All tags in the panel.

If, for example, you wanted to copy all .COM files, you would put the cursor on any .COM file, press Alt-E, and C. Or suppose that at the end of the day you wanted to copy/update all the files written or revised today. Just put the cursor on any file with today's date and press Alt-D and C. Then suppose you wanted to delete all the earlier files. Press A to clear the tags, Ctrl-O and D, and confirm the deletion as requested after a final check of the screen.

### Copy with rename

You may copy a file and rename it in one operation. Only one file at a time may be copied in this way. The command is Alt-R (copy with Rename). It operates only on the file at the cursor.

### The file freshener

Suppose you have a floppy disk with backups for some of the files in a hard disk directory, and you want to freshen them periodically. With either panel set up with the primary directory whose files you are backing up, put the backup floppy in a floppy drive, arrow across to the other panel and press N and the drive letter to bring up its root directory (now in the source panel), and then just press Shift-F (for Freshen). CF tags all the backup files in the source panel if none were already tagged or leaves existing tags as is if some were tagged. Then CF tags all the files in the opposite panel with the same name as tagged files in the source but which are NEWER, then copies these to the source, and finally updates the source panel display leaving the primary directory in the target panel display tagged to show you what was copied. Just four keystrokes! No excuses ever again for not backing up files.

### Different ways to show the files

Ctrl-O (for *Order*) sets a screen which gives seven choices for file ordering with the cursor blinking on the current selection. Press a number key 1 through 9. The files will now be reordered in the new scheme. The options for ordering by date/time are particularly useful in the visual sense when backing up just the files revised today as in the example above.

### The data entry window

The data entry window you encountered in the renaming exercise responds to most of the usual line-editing keypresses.

- Insert toggles the mode between Typeover and Insert. The mode stays set for each subsequent entry. In Typeover mode, any default entry is cleared if the first keystroke is an alphanumeric character.
- Ctrl-Lf/Rt Arrow and Tab/Shift-Tab go right or left to the space following the next blank or punctuation mark. Shift-Lf/Rt Arrow go to beginning or end of data field. Home goes to the beginning of the field.
- Lf/Rt Arr, Bksp, and Del perform the usual functions.
- End goes to the blank after the last non-blank character.
- Alt-Keypad permits entry of any ASCII code as a decimal number. Hold down the Alt- key while you type in a number from 1 to 255 on the numeric keypad. When you lift the Alt- key, the IBM symbol for that ASCII code will appear in the window and the cursor will advance a space.
- Ctrl-D deletes to the end of the line.
- Esc aborts the operation.
- Down Arrow is equivalent to Enter. In some cases, Up Arrow moves up a line. (The rename facility in the main module and the replace string facility in the editor use this convention.)
- For entering filenames and subdirectories, all letter keys are registered as uppercase, regardless of Caps Lock or Shift-key positions, just for the sake of uniformity and ease of alphabetizing. For command-line parameters, which may be case-sensitive, both cases are enabled.

### Disk Space Occupied versus Actual File Size

Now take a look at the file listing. You see the name of the file, date, time and size in K. As with the disk space information just below the file list display, size is in integer multiples of clusters occupied. If you want to see exactly how big a file is in bytes, press + to expand the size field. (I had to give away the file time to do this and still keep the two panels readable.) Shift back to the contracted kilobyte form of file size with -.

### File attributes

Also in the file list, to the far right in each panel, you will probably see A's. This means that the

DOS Archive attribute bit is set in the file attribute byte. DOS sets this bit every time it operates on a file. CF lets you operate on this bit and the other bits in the DOS file attribute byte — Read-only, Hidden, and System. Before doing this exercise, look at the top line of the display. If you see *Hide=N*, that means that files with the DOS Hidden attribute set will be displayed anyway — i. e., the CF Hide switch is off. Chances are you will see Y instead of N, meaning that the Hide switch is on the default setting. In this setting, files will disappear from view as you set the DOS hidden attribute, so we need to toggle the hide switch to N. Press Ctrl-H to toggle the hide switch off.

Either tag one or more files, or position the cursor on the file whose attribute(s) you want to set, and press 1 to toggle the state of the Read Only attribute, 2 to toggle the Hidden attribute, 3 to toggle the System file attribute, 4 to toggle the Archive attribute, or 0 (zero) to clear all attributes. You may also toggle the hide attribute (with 2) of a subdirectory, but this must be done one subdirectory at a time. Note that hiding a file makes it invisible to CF only when the hide switch is set on, as shown in the top line. Toggle this hide mode off and on with the combination Ctrl-H (discussed further below). Decide for yourself which mode you prefer. Some people like to hide the overhead files and directories on their disks and leave the hide switch on as the default setting to clean up the display.

### Changing the date/time stamp of a file

You may change the date and/or time of a file by putting the cursor on it and pressing Alt-F. A data window opens for you to first enter the new date, using the current date in the file's date/time stamp as the default, and then the time. This operation is not permitted on Read-only files.

### Alt-Tagging and appending

Suppose you have two files that you want to stick together as one heel-to-toe. Tag them with append tags with the key combination Alt-T or Alt-Space in the order in which you want them to be concatenated, and press C. CF will offer you a filename for the new concatenated file consisting of the name of the first file Alt-Tagged plus the extension .APF (for APpended File). You may edit or accept that name as given. Then it creates this file in the target path and appends into this file each of the Alt-Tagged source files in order. You may concatenate up to 35 files at a time this way. The order in which the file was Alt-Tagged is shown in the character that appears to the left of the filename as it is tagged (1–9, then a–z). If you had pressed B instead of C after affixing the Alt-Tags, the concatenated file would have been written as a backup into the source path instead of the target.

### Printing a file

You can print a file to the parallel printer just by putting the cursor on the file and pressing L (print fiLe). The file will be put into a special print queue for printing to line printer 1 on the parallel port as a background process while you are doing other things, like editing another file, updating disks, etc. Up to five files may be put in the print queue, which may be viewed with Shift-L.

At times the printer may halt momentarily during disk operations. CF gives preference to disk operations over printing to avoid any conflict in time-critical operations. You may terminate printing with Ctrl-L. This actually clears the entire print queue. To force a form feed at the end of the file you have just queued, press Ctrl-F before you queue the next file. CF sets an internal flag to check that the last character sent to the printer from that file is a form feed. If it is not, then it sends one. (Ctrl-F is active when no file is printing, also, as a way of form-feeding the printer from the keyboard.)

## CMFiler 5.35 at a Glance

The following is an overview of CMFiler's commands in various modules.

### Disk/directory operations in Main Module:

| | |
|---|---|
| Alt-Q or Esc+other key | Quit CMFiler |
| N | New drive or disk in source panel |
| V | specify Volume label |
| Shf/Ctl-N | view/edit Notes |

| | |
|---|---|
| \ | display root directory in source |
| Enter | display dir in source panel |
| Shf-Enter | display dir in targ panel |
| P | display Parent in source panel |
| Shf-P | display Parent in target panel |
| M | Make directory in source panel |
| D | Delete directory (must be empty) |
| Alt-L | print directory Listing |
| R | Rename directory |
| H | Help |
| Shf-T | go to Tree module |
| Shf-O | setup Options |
| Shf-S | Save option switches |

## File operations in Main Module:

| | |
|---|---|
| 1234 | toggle file attrs RHSA |
| T or Space | normal file Tag |
| Alt-T or Alt-Space | Alt Tag |
| A | set/reset All tags |
| Ctl-A | toggle All tags |
| Alt-E | tag all files with same Ext |
| Alt-P | replicate tags in oPposite pnl |
| Alt-F | change File date/time |
| Alt-N/O/D | tag all Newer/Older/same Date |
| C | Copy/append file(s) to target |
| Shf-C | move files within disk |
| B | copy/append file(s) as Backup in source panel with extension "._xx" |
| R | Rename file |
| Alt-R | copy file with Rename |
| D | "soft" Delete file(s) in source |
| Ctl-D | "hard" Delete file(s) |
| F | create new File |
| Shf-F | Freshen source from target |
| E | Edit existing file |
| Enter | view existing file, no edit |
| L | print fiLe with spooler |
| Shf-L | look at fiLe print queue |
| Ctl-L | clear fiLe print queue |
| Ctl-F | Form feed line printer |
| G/Alt-G+ltrs | Go to file/dir |
| Shf-L/R Arr | go across to match file |

## Display enhancements in Main Module:

| | |
|---|---|
| Ctl-E | select alphabetic casE |
| Ctl-P | select color Palette |
| Ctl-C | toggle Compare mode Ctl- |
| H | toggle Hide mode |
| Ctl-O | select file Order scheme |
| Ctl-S | force screen-Saver |
| Ctl-M | set file Mask |
| +/– : | long/short form of file size |
| 5 | toggle system date/time display |

## Directory operations in Tree Module:

| | |
|---|---|
| N | New drive or disk in source panel |
| P | move cursor to Parent directory |
| M | Make directory under cursor |
| R | Rename directory at cursor |
| 2 | hide directory |
| Ctl-H | toggle Hide switch |
| Ctl-O | Toggle file Overcopy switch |
| Ctl-C | toggle Compare switch |
| Shf-L/R Arr | go across to path match |
| Shf-U/D Arr | go to next dir in level |
| C | Copy source under target |
| Shf-C | move source under target |
| Alt-C | Copy just directory array |
| I | copy source Into target |
| Shf-I | move source Into target |
| J | copy Just files in directory |
| Shf-J | move Just files |
| D | "soft" Delete source structure |
| Ctl-D | "hard" Delete source struct |
| Alt-D | "hard" delete just files |
| Alt-L | Lprint tree structure |
| S | Show files |
| Alt-R | Refresh tree from disk |
| G/Alt-G | Goto file/directory |

| | |
|---|---|
| Shift-F | Find text |
| Esc | return to main, old paths |
| Enter | return to main, new paths |
| H/F1 | Help |
| Alt-Q | Quit CMFiler |

## File viewing/editing

| | |
|---|---|
| F1 | Help |

### Alt- - modified keys:

| | |
|---|---|
| S | Switch from view to edit |
| F | Find string |
| R | search and Replace string |
| G | Global search and replace string |
| X | find/replace neXt matching string |
| M | Mark beginning or end of block |
| C | Copy marked block |
| V | moVe marked block |
| Y | delete block |
| P | Print marked block |
| O | Output marked block to a file |
| Keypad number | ASCII code entry direct from keypad |
| N | type Null character |
| +/– : | enable/disable delimiter markers |
| W | toggle line Wrap mode |
| H | Help |

### Ctrl- - modified keys:

| | |
|---|---|
| T | delete word |
| D | delete to end of line |
| Y | delete line |
| F | Form feed line printer |
| Enter | select delimiter type (CR,LF,CR+LF,LF+CR) (in edit mode) |

### Cursor moves:

| | |
|---|---|
| Esc | exit |
| Enter in view mode | exit |
| Unmod Arrows | move one character or line |
| Ctl-Lf/RtArr | move to first character of next word, left or right |
| Shf-Lf/RtArr | move to beginning/end of line |
| Home/End | move to beginning/end of line |
| PgUp/PgDn | move one video page (20 lines) |
| Ctl-PgUp/PgDn | move 10 video pages (200 lines) up or down |
| Alt- B/E | move to Beginning/End of file |

### Child process operations in Main Module:

| | |
|---|---|
| S | Shell to DOS |
| K | set resident Kernel size (L or S) |
| Q | Quick execute as child process |
| Ctl-Enter | set comm tail delimiter |
| Alt-C | add file spec to Comm tail |
| Alt-T | Tag for addition to comm tail |
| X | ready file spec for eXecution — follow with Alt-/Shf-/Ctl-X: |
| Alt-X | add Alt-Tagged files to comm tail, edit comm tail, eXecute on Enter |
| Shf-X | add Alt-Tagged files to comm tail, no edit, immediate eXecute |
| Ctl-X | add file at cursor to comm tail, no Alt-Tagged files, no edit, immediate eXecute |
| Shf-F10 | add/edit custom F-keys |
| F10 | view/select F-keys |
| F1-F9 | ready application assigned to custom F-key for execution, add default offering to comm tail — follow with Alt-/Shf-/Ctl-X |
| Shf-F1-F9 | ready F-key, add default comm tail offering and Alt-Tagged files to comm tail, immediate execute |
| Ctl-F1-F9 | ready F-key, add default comm tail offering and filename at cursor to comm tail, no Alt-Tagged files, immediate execute |
| Z | ZIP Alt-Tagged files in source to Alt-Tagged target, with comm tail edit |
| Shf-Z | Same, without comm tail edit |
| U | UNZIP file at cursor to target, with comm tail edit |
| Shf-U | Same, without comm tail edit |

## CMFiler 5.35 Registration Form

Name: ______________________
Company: ______________________
Address: ______________________
City: ______________________
State: ________ Country: ______________
Zip: ________ Phone: ______________

Make checks payable to Charles F. Martin or NoVaSoft and

**Remit to:**
3239 Riverview Dr.
Triangle, VA 22172-1421 USA
Phone (703) 221-1471/1833 CompuServe 72130,1400
Internet 72130.1400@compuserve.com

Enclosed is $30 to register as a user of CMFiler. I understand I will receive by mail a copy of the latest registered version of CMFiler and a hard copy of the user's manual. I will also receive a free copy of a future major upgrade, as well as user support by mail for at least one year. CompuServe subscribers may receive their user support by CIS E-mail, and periodic free upgardes for as long as they wish.

In extension of the single-user license conferred to my company by this registration, also enclosed is $_____ for _____ additional complete copies of CMFiler at $15 per copy, and/or $_____ for _____ unsupported additional users at $7.50 per additional users ($5 per additional student user at schools), based on a good-faith estimate of the total number of users afforded access to CMFiler as a workplace resource. I will inform users that the license extensions thus granted do not authorize their private use of CMFiler outside my company's workplace environment.

Send my upgrades on ☐ 5.25" ☐ 3.5" floppy disk.

Also send me upgrades by E-mail.
My CIS ID is ______________________

I also understand I will receive a $10 commission every time a new registrant refers to me as the source of his/her evaluation copy of CMFiler. The registered user who gave me my shareware copy of CMFiler, and who will be sent a $10 commission is

______________________
Name of Registered User/Street/City/State/Zip

Other ways to register:

* With Master Card, Visa, American Express, or Discover from the Public (software) Library (PsL), by voice to 1-800-2424-PsL or 713-524- 6394, by FAX to 713-524-6398, or by CompuServe Email to 71355,470. Order CMFiler, part number 10657. These phone numbers are for orders only. To ensure that you get the latest version, PsL will notify us the day of your order, and we will ship the product directly to you. Any questions about the status of shipment of the order, refunds, registration options, volume discounts, dealer pricing, site licenses, etc., must be directed to NoVaSoft at 703-221-1471 or the address above, and not the PsL order numbers. You may also mail credit card orders to PsL at P. O. Box 35705, Houston, TX 77235-5705.

* With major credit card through the CompuServe Shareware Registry. Just GO CIS:SWREG and order entry #15.

# Disk OrGanizer

**Version 3.17**

**Copyright © 1991 by Soft GAM's Software**

## Introduction

A disk that has been in use a while will usually become slower accessing the data and programs that have been stored on it. This is due to two problems:

- The disk has become fragmented.
- There are deleted entries in directories.

Disk OrGanizer will solve these problems and make your disk run like new. If the size of a directory decreases enough after removing the deleted entries, Disk OrGanizer can also recover some disk space.

Disk OrGanizer has been tested with MS-DOS 2.10, 3.01, 3.10, 3.30, 4.01, and 5.0 (and by Bob on 6.0) on logical drives as large as 600 megabytes. It is believed that Disk OrGanizer will operate on any MS/PC-DOS disk.

Disk OrGanizer performs extensive tests before it starts to move data to make sure that the File Allocation Table (FAT) and directories are in good repair.

## Program and documentation files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the DOG files in the directory \SECRETS\DOG. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the DOG files to a directory that is included in the search path.

## Installation

To install Disk OrGanizer, you need only copy it to a directory on your PATH.

***Important:*** Please back up your hard disks before you run Disk OrGanizer.

You can test Disk OrGanizer by typing the following command:

```
DOG [d:] /TEST
```

*d:* is an optional drive specifier. (The brackets mean optional and should not be typed.) If no drive is specified, the current drive will be used.

If Disk OrGanizer reports errors and you have not run Disk OrGanizer before, you should run CHKDSK with the /F switch and repair the disk. If you have not backed up your disk, Disk OrGanizer will warn you as to how many files may need to be backed up.

Disk OrGanizer will also list any files that are open on the disk that you are organizing. Disk OrGanizer will not be able to move these files nor pack the directories that they are in. If you have used other disk defragment programs before and have had no problem, it should be safe to run Disk OrGanizer. Disk OrGanizer will move HIDDEN files; most other disk defragmenters don't move these files. If you have not run any disk defragment programs before, you should read the rest of this chapter.

## Background programs

Disk OrGanizer reads and writes to the disk directly. If a program tries to write to the disk while Disk OrGanizer is operating, data may be lost. You should not run Disk OrGanizer on a disk that you know will be written to. Programs that are likely to write to the disk are printer spoolers, multitasking programs like Windows or DesqView, switching programs, and network programs.

If you have FASTOPEN installed on your computer, you will need to reboot your computer after you run Disk OrGanizer. Some background programs leave files open all of the time. Disk OrGanizer will not move files that are open, and it will not pack a directory that has open files in it. If you do not know if you have these programs on your computer, I would suggest that you run Disk OrGanizer from a floppy disk as described in the "Quick Start"section.

## Copy-protected programs

Disk OrGanizer may make some copy-protected software unusable. It is recommended that you remove any copy-protected software from your disk before you run Disk OrGanizer if you can't determine its compatibility with Disk OrGanizer.

## Power failure

Disk OrGanizer can recover from a power failure or reboot. If a power failure or inadvertent reboot occurs while Disk OrGanizer is running, you should run Disk OrGanizer again as soon as possible. Disk OrGanizer will report any errors that it finds on the disk and ask you if you would like to repair them. If you answer yes to all of the questions, Disk OrGanizer will repair the disk and then continue operation as normal. You should not use any third-party CHKDSK-type programs that recover lost clusters. If you use CHKDSK, you should NOT make the lost cluster chains into files.

## Quick start

If you would like to defragment your disk and get back to work without having to read the rest of this manual, the simplest and safest way to run Disk OrGanizer is described here. The first thing to do is to find an old or unused floppy disk. Then format this disk with a copy of the system:

```
FORMAT A: /sys
```

Copy DOG.EXE to the floppy disk:

```
COPY DOG.EXE A:
```

You may need to create or copy CONFIG.SYS on the new disk if you have disk drivers or an EMS memory driver that needs to be installed.

Reboot your computer **(Ctrl-Alt-Delete)** with the new disk installed in the A: drive.

DOS will ask you for the date and time. This is because there is no AUTOEXEC.BAT file. Enter the correct date and time.

Run Disk OrGanizer on your hard disk.

```
DOG C: /AUTO /CRC /TRUNCATE
```

Disk OrGanizer will test the disk (like CHKDSK), give you a report, and analyze the disk to determine the best place to have the directories and files on the disk. Disk OrGanizer will tell you how many directories need to have deleted files removed (packed), tell you how many directories can be made shorter (truncated), and tell you the number of clusters that it will have to move to defragment the files and directories on the disk.

You will then be asked if it is OK to write to the disk. If you answer no to this question, Disk OrGanizer will quit without ever having written to the disk. If you answer yes, DiskOrGanizer will pack the directories, truncate the directories, and then start moving clusters.

While Disk OrGanizer is moving clusters, you will see the following message:

```
X nnnnn - elapsed time 0:00:00
    estimated 0:00:00
```

*X* is either R)eading, S)orting, C)omputing CRC, W)riting, T)esting CRC, L)inking new clusters, or U)pdating FAT and directories. *nnnnn* is the number of clusters that have been moved. *Elapsed time* is the amount of time Disk OrGanizer has been moving clusters, and *estimated* is the approximate amount of time it will take to organize the disk.

You can quit from Disk OrGanizer at any time by pressing the escape (Esc) key or a ^C. Stopping Disk OrGanizer while it is moving clusters may cause extreme fragmentation. Rerun Disk OrGanizer as soon as possible if you abort while it is moving clusters. If you think that Disk OrGanizer has stopped running, press the spacebar. You should then hear a beep within a few seconds. When it is done it will check the disk once again, give you a new report, and then exit.

You should then remove the floppy disk from the A: drive and reboot your computer. You should notice that the programs on your disk load a little faster. If Disk OrGanizer truncated some directories, you will also have more free disk space.

## Using Disk OrGanizer

The purpose of Disk OrGanizer is to speed up the access of files on your disks. There are two things that slow down access of data: deleted files in directories and fragmented files. The actual placement of files on a disk has very little to do with the amount of time it takes to load a file. Most disk defragment programs take a long time to run, and this keeps people from running them often.

Disk OrGanizer has complex algorithms that allow it to defragment all of the files on a disk while moving very few clusters. Although Disk OrGanizer can control the placement of files and directories on the disk, I would suggest that you refrain from using this feature as it can greatly increase the number of clusters that Disk OrGanizer will need to move and give you very little benefit. The exceptions are directories that are read often, such as the directories on the PATH which you may want to move near the File Allocation Table (see the AUTO command.)

### Modes

The mode determines how Disk OrGanizer will organize a disk:

| | |
|---|---|
| FILL | Defragments files and puts them as close to the FAT as possible. FILL will not leave any free space between files. FILL may leave files fragmented around BAD disk sectors and unmovable files. This mode gives you all of the benefits that you get from other disk defragmenters but only takes a fraction of the time to run. |
| FAST | Defragments all files and attempts to put them in one area of disk space. FAST may leave gaps of free space between files. |
| QUICK | Like FAST mode but does not try as hard. |

| | |
|---|---|
| DEFRAG | Defragments the files on the drive without concern for their placement. |

The following modes will more than likely cause all of the files and subdirectories on a disk to be moved.

| | |
|---|---|
| DATE | Puts the files on the disk, oldest files first (closest to the FAT). Using this mode once in a while, it can take a long time to run, may decrease the number of clusters that Disk OrGanizer will need to move with the FAST or FILL mode. |
| DIRECTORY | Puts the files on the disk in the order it finds them in the directories. That is, the first file or subdirectory in the root directory will be closest to the FAT (also referred to as LOW), followed by the second file in the root, etc. |

The following modes are for compatibility to older versions. They are of little use.

| | |
|---|---|
| FRAGMENTED | Puts all of the files that are not fragmented on the disk first, and then puts the fragmented files on the disk. Using this function will slowly put all of the files that change size away from the FAT. This mode does not work well if you have BAD clusters or system files other than DOS on the disk. |
| PACKED | Leaves the files in the same order as it finds them on the disk, by cluster. |

## Switches

| | |
|---|---|
| CRC | Causes Disk OrGanizer to read data after it is written to the disk, and tests to make sure that it was written correctly. |
| BATCH | Tells Disk OrGanizer to suppress all prompts. This is useful for running Disk OrGanizer from a batch file. If this switch is set, any error will cause Disk OrGanizer to terminate. |
| NOVERIFY | Tells DOS not to verify data written to the disk (see VERIFY in your DOS manual). This will increase the speed of Disk OrGanizer, but will allow data to be lost. This switch should not be used on a floppy disk, but it may be used on a hard disk at your own risk. This is for people to whom speed is more important than data integrity. If you have set the CRC switch, then Disk Organizer will verify the data, and you can include this switch to improve performance. |

## Command-line syntax

The command-line syntax to run Disk OrGanizer from the DOS prompt is as follows:

```
DOG [d:] [orderfile] [/commands]
    [/mode] [/switches]
```

(The [ ]'s mean optional and should not be typed.)

*d:* is an optional drive specifier. If no drive is specified, the current drive will be organized.

*Orderfile* is the path and filename of the order file. If this is not included, Disk OrGanizer will look for a file called ORDER.DOG in the root directory of the drive being organized, and if it does not find that, it will look in the environment for DOG=d:\path\filename and use that file. If no orderfile is found, the DOS NUL device is used. If you have an orderfile and don't wish to use it, you can tell Disk OrGanizer to use the NUL file as an orderfile (i.e., DOG NUL /AUTO).

Commands consist of AUTO and TRUNCATE.

Mode tells Disk OrGanizer where to place the files on the drive. You must supply a mode to suppress the shareware sign-on message. The mode can be set on the command line or in an orderfile.

Switches allow you to control several options as to how Disk OrGanizer will test the data that is moved and interact with the operator. Switches can be set on the command line or in an orderfile.

## Command-line commands

| | |
|---|---|
| TRUNCATE | Tells Disk OrGanizer to truncate all subdirectories. |
| AUTO | Tells Disk OrGanizer to put the directories defined in the PATH environment variable first, followed by the file defined by the COMSPEC environment variable, and then all of the other directories. If no other mode has been specified, it will then order the rest of the files as if the /FILL mode was set. |

## Orderfile syntax

The orderfile can be used as a simple configuration file or can be used to tell DOG the exact order that you would like to have your files on a disk.

Modes and switches that are given on the command line will override those in the orderfile.

You may find it more convenient to run Disk OrGanizer from a batch file than to use the orderfile feature. Again, I would like to point out that the placement of files has little effect on the speed at which they can be read.

If you use the AUTO command-line command, you may not end up with what you expected. (See listing 2.1.)

## Setting the mode

To give a mode in the order file, enclose it in square brackets ([]). A mode must be given on the command line or in the order file for Disk OrGanizer to run. A mode given on the command line will override the mode in the ORDER file.

## Setting switches

Listing 2-1: A very simple ORDER.DOG file

```
               ; The []s must be
               ; included.
[FILL]         ; set the mode
               ; Truncate all of
               ; the directories
[TRUNCATE ALL] ; Directories that
               ; that have already
               ; been specified,
               ; will not be
               ; truncated. The
               ; AUTO command line
               ; command appends
[PATH]         ; these three
               ; orderfile commands
[COMSPEC]      ; to the end of the
               ; orderfile.
[ALLDIR]       ;
;
```

## Setting switches

Switches can be specified in the orderfile by enclosing them in square brackets ([]). (See the section on switches above.)

### Order file commands

The area of the disk close to the FAT is called the LOW area and the area away from the FAT is called the HIGH area. Any file that is not named anywhere in the ORDER.DOG file is said to FLOAT.

In making this file, you should name as few files as possible.

The following three commands are a shorthand for file and directory names:

| | |
|---|---|
| PATH | The same as listing all of the directories in the PATH environment variable that are on the drive being organized. |
| COMSPEC | The filename given in the COMSPEC environment variable is used if it points to the drive being organized. |
| ALLDIRECTORIES | The same as listing all of the directories on the drive. |

The following commands tell where to place named files:

| | |
|---|---|
| LOW | Tells Disk OrGanizer to place the files described by following filenames close to the FAT. This also affects the FREE command. |
| HIGH | Tells Disk OrGanizer to place files described by following filenames away from the FAT. This also affects the FREE command. |

Listing 2-2: Sample ORDER.DOG file

```
;
[DEFRAGMENT]    ; if no mode is
                ; given on the
                ; command line use
                ; DEFRAGMENT
[FORCE]         ;
\COPYPRO.SYS    ; I don't want this
                ; file to move
[FLOAT]
[TRUNCATE]      ;
\DIR_A\         ; Truncate sub-
                ; directory DIR_A,
                ; but let it FLOAT.
;
[LOW]           ; Set move files
                ; LOW.
\DOS\           ; Truncate sub-
                ; directory DOS, and
                ; put it
                ; next to the FAT.
[ENDTRUNCATE]
\COMMAND.COM    ; COMMAND.COM will
                ; be placed after
                ; \\DOS.
[FREE 500]      ; leave some room
;
[HIGH]          ; set move files
                ; HIGH.
\AUTOEXEC.BAT   ; As this file is
                ; used only once, on
                ; reboot, we can
                ; move it away from
                ; the FAT.
\CONFIG.SYS     ; another seldom
                ; used file.
\ORDER.DOG      ; This file will be
                ; placed just in
                ; front of
                ; CONFIG.SYS
```

| | |
|---|---|
| FORCE | Tells Disk OrGanizer not to move the files that follow. If you are going to FORCE files, you should do it before you place files LOW or HIGH. |
| FLOAT | Tells Disk OrGanizer to ignore the files that follow. This is used if you wish to TRUNCATE a subdirectory but don't want to put it LOW, HIGH, or FORCE. ***Note:*** In future releases this may be used to allow SYSTEM files to be moved. |

These commands control whether named directories will be truncated.

| | |
|---|---|
| TRUNCATE ALL | Truncates all directories on the disk. |
| TRUNCATE | Subdirectories named between TRUNCATE and ENDTRUNCATE will have their size changed so that they will use as little space as possible. |
| ENDTRUNCATE | See TRUNCATE above. |

System directories and directories that have already been placed LOW, HIGH, or FORCED will not be truncated.

The following command will force clusters to be left free.

| | |
|---|---|
| FREE nnnn | Tells Disk OrGanizer to leave *nnnn* (a number) clusters of free space after the last file named in the order file if files are being placed LOW or before the file if files are being placed HIGH. Disk OrGanizer ignores this command if there isn't enough free disk space. The *nnnn* may be replaced with an asterisk (*) if you want all available free clusters in a location. |

## File identifiers

A file identifier is a string that starts with a backslash (\), has zero or more subdirectory names separated by backslashes (the Path), and ends with a filename or subdirectory name. A file identifier ending with a subdirectory name can be followed by an optional backslash.

It is important to understand that a subdirectory is a file with a special format and that Disk OrGanizer treats subdirectories just like files in the orderfile except that subdirectories can be TRUNCATED. You must always include the full path of a file or subdirectory. Wildcards may be used in filenames. But please note that including a large number of files in the ORDER file can greatly increase the amount of time it takes Disk OrGanizer to organize a disk.

### Comments

A comment starts with a semicolon and ends with a carriage return.

## Registration, Copyright, and Technical Support

This document and the program files DOG.EXE ("the software") are copyrighted by the author. The copyright owner hereby licenses you to use the software, given these restrictions:

- The program shall be supplied in its original, unmodified form which includes this documentation.
- For-profit use without a license is prohibited.
- The program may not be included or bundled with other goods or services. Exceptions may be granted upon written request only.
- No fee is charged; an exception is granted to not-for-profit user's groups, which are permitted to charge a small fee (not to exceed $5) for materials, handling, postage, and general overhead. No other organization is permitted to charge any amount for distribution of copies of the software or documentation, or to include copies of the software or documentation with sales of their own products.

Technical support is available by phone, by mail, CompusSrve, and E-mail to registered and licensed users. Limited support is also available to help you to get Disk OrGanizer to operate on a computer so that you can evaluate it.

CompuServe: PPN# 73210,3374

the Bit Boutique BBS 707-778-8944, 2400 baud

Leave E-Mail to ALLEN MORRIS

### DOG Registration Form

Name: ____________________

Company: ____________________

Address: ____________________

City: ____________________

State: __________ Country: __________

Zip: __________ Phone: __________

**Remit to:**

G. Allen Morris III
Soft GAM's Software
P.O. Box 1311
Mendocino, CA 95460
SSN# 559-29-3621
(707) 961-1632 (10 a.m. to 6 p.m. Pacific Time)

No license is required to run Disk OrGanizer on a computer that is not used for business. The fee for a license depends on the estimated number of copies of the program that you will use. For local area networks, one copy must be licensed for each computer that will be using the program on the network. A license is not transferable.

| **Quantity** | **Price** | **Update** |
|---|---|---|
| 1 | $30.00 | $6.00 |
| 2 to 9 | $25.00 | $5.00 |
| 10 to 49 | $20.00 | $4.00 |
| 50 to 99 | $15.00 | $3.00 |
| 100 to 199 | $10.00 | $2.00 |
| Unlimited at one sight | $2,000.00 | $400.00 |
| Additional sites unlimited at each sight | $1,000.00 | $200.00 |

The price to update to a new version applies only to the quantity previously registered. Regardless of the quantity purchased you will receive only one disk and one printed manual. Additional manuals are available for $6 each. You may make as many copies as are licensed and one archive copy. Orders of $50 or less must be prepaid. Orders over $50 — NET 30 days.

# DUPE

**Version 1.23**
**Copyright © 1992, BEST TECHNOLOGY**

## Introduction

DUPE is a shareware utility that allows you to copy disks faster and easier than MS-DOS's DISKCOPY program.

**DUPE's advanced features:**

1. Copies an entire floppy disk, any size, in a single drive with *no disk swapping*.
2. Can write multiple copies of a disk while reading the original only *once*.
3. Copies only the *used* portion of a disk.
4. Beeps after each task.
5. Formats disks up to 50 percent faster than DISKCOPY.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the DUPE files in the directory \SECRETS\DUPE. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the DUPE files to a directory that is included in the search path.

## Command Format

The program should be invoked as follows:

```
DUPE [drive]: {Options}
```

*[drive]* is a mandatory argument specifying the disk drive to use for the copy operation. *Options* are as follows:

| | |
|---|---|
| /W | copy Whole disk |
| /U | copy Used portion of disk only |
| /FN | Format destination only when Necessary |
| /FA | Format destination Always |
| /WO | Warn before Overwriting |
| /NW | overwrite with No Warning |
| /S | Sound on |
| /Q | Quiet—no sound |
| /C | Clone options to EXE |

## Registration

To register, perform a <Print-Screen> of the on-line order form, fill it out, and sent it to BEST TECHNOLOGY.

### DUPE 1.23 Registration Form

Name: ____________________
Company: ____________________
Address: ____________________
City: ____________________
State: __________ Country: ____________________
Zip: __________ Phone: ____________________

**Remit to:**
BEST TECHNOLOGY
915 Sherwood Forest Lane
Saugus, MA 01906, U.S.A.

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Payment must be made in checks drawn on U.S. Banks in U.S. Dollars, or CASH.

DUPE Single User version. Payment must be enclosed. Single user version must be PREPAID.
each ........ $12.95 $ ______
DUPE Corporate User version
each ........ $29.95 $ ______
DUPE Site License version
each ........ $99.95 $ ______
*Corporate and Site License orders are billed if Purchase Order is enclosed.*
MASS Sales/Use tax (MASS residents only)
........ 5% $ ______
Surcharge for Foreign Orders .... $10.00 $ ______
Shipping and Handling ........ $3.00 $ ______
TOTAL ........ $ ______

# ExtraDOS Toolbox Professional

Copyright © 1985–1992 by Foley Hi-Tech Systems

## Overview

ExtraDOS Toolbox Professional is a compilation of powerful and sophisticated tools for MS-DOS users. The programs offer functionality. Some enhance programs already found in MS-DOS; other tools were familiar to the Unix world and are now available in MS-DOS. ExtraDOS also compliments PC Tools and Norton utilities by offering many features not found in these packages. The quality of our tools will allow you to work faster, smarter, and better.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the ExtraDOS files in the directory \SECRETS\EXTRADOS. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the ExtraDOS files to a directory that is included in the search path.

## Features Of ExtraDOS Toolbox

The following documents that appear bold are included on the enclosed diskettes. All other documents are available from Foley Hi-Tech Systems.

| | |
|---|---|
| ADDCOMM | Allows you to define the actual address of your communications ports in the DOS lower memory segment, allowing older machines to support COM3 and COM4. |
| **ALARMCLK** | **Provides a simple alarm clock for MS-DOS machines. Used with SETALARM.EXE.** |
| **ALLSUB** | **Allows any command or program to be run on the current directory as well as all on subdirectories of the current directory. This is useful for deleting groups of files or performing tasks with programs that don't normally allow for subdirectories.** |
| AREACODE | AREACODE is a utility that will help you locate the region to which an area code applies or to locate the area code for a specific region. |
| BANNER | Create a large banner using a standard ASCII printer. |
| BOOT | Reboot PC from within a batch file or from the DOS command line. |
| BOOTCTL | Boot system on hard drive even with a non-system floppy disk in the A: drive. |
| BOOTLOCK | Locks out the [CTRL]C, [CTRL][BREAK], and [CTRL][ALT][DEL] keystroke functions. Disable one or any combination of these functions. |
| BRKBOX | Software version of a breakout box. Displays the status of the Serial or Parallel ports on the screen. This is a very handy tool when trying to debug communications or printing problems. |
| **CALENDAR** | **Display monthly or yearly calendar.** |
| CAT | UNIX-like CAT utility for displaying, printing, and combining files. |
| CHIMES | Quarterly hour clock chimes. Westminster, Saint Michael, or Whittington chimes. |
| CLEANFL | Makes the task of cleaning floppy drive heads much easier and more thorough. Moves the head of the floppy drive to travel the entire span of the cleaning floppy rather than just using a small portion of the cleaning diskette. |

| | |
|---|---|
| **CLEANUP** | **Cleanup allows you to recover valuable space that is eaten up by unused or duplicate files on your hard and floppy drives. Quickly isolates unused and duplicate files and removes them.** |
| COUNT | Counts the number of characters, words, and lines in a file. |
| CREDITS | A listing of those responsible for ExtraDOS Toolbox Professional. |
| CURLOCK | CURLOCK locks the cursor shape (optionally set with CURSOR) so that no matter what software you use the cursor will not be changed. |
| CURSOR | Lets you change the size and shape of the DOS cursor. |
| CUT | UNIX-like utility for displaying and printing text files in different formats by eliminating columns and other formatting options. |
| CUTPASTE | Small clipboard utility for MS-DOS. CUTPASTE allows you to pop up over any DOS application and copy a portion of the information displayed on the screen and paste it into another application, to the printer, or to a file. |
| DELAY | Sets a user-defined delay in batch files or CONFIG.SYS. |
| DETAB | Strips any tab characters from an ASCII text file and converts them into spaces. This is especially useful when working with old formatted files that you would like to work with in an editor which treats fixed tabs differently. |
| **DISKINFO** | **Reads the boot sector of a disk and then makes a DOS call to find out the information that DOS has for that disk. It will then list out the information in a side-by-side format so that you may compare the reports from DOS and the boot record.** |
| EDIT | SuperEdit full-screen editor. Many features including multiple windows, text-formatting options, and pull-down menus. |
| EVAL | DOS command-line calculator that evaluates mathematical expressions input on the command line. |
| FILEATTR | Allows you to set or change the attributes of a group of files. |
| FILEINFO | Alleviates the problem that DOS has created by only allowing 8 characters to describe a file. FILEINFO provides a way of adding comments to files in any directory. |
| FILESIZE | Displays a list of files and their respective sizes with summary that gives total sizes, number of files, total space on disk, and space available both in bytes and percentage. |
| FILETREE | Scans the directory structure of a disk and gives a top-down tree listing of the entire disk's directory structure. |
| FINDFILE | Will locate files by searching the entire disk for a specified file. Also searches within compressed files to search for filenames. When it finds a matching file, it will display the directory the file was found in, the complete filename, the file size, creation date and time, and any attributes associated with the file. |
| HELP | HELP prints out help messages on some of the more useful and popular DOS commands. |
| HEXDUMP | Provides you a quick dump of the contents of any file. The advantage of HEXDUMP is that you can quickly dump the contents of any file to the screen without having to |

| | |
|---|---|
| | load the overhead of a shell program. |
| HUSH | A utility that can eliminate a majority of the sounds generated by the PC's internal speaker. |
| KILLDIR | Makes eliminating a directory and all of its contents a lot easier. It displays information about the directory that includes the number of files and their sizes as well as any subdirectory information. |
| LASER | Helps you set your laser printer's mode and also makes it easy for you to send control codes to it. |
| **LICENSE** | **License information.** |
| LS | UNIX-like file-listing utility. It has many advanced features beyond that of the DIR command found in DOS. |
| MONITOR | MONITOR displays an adjustment pattern for your monitor. It is useful when trying to align and adjust monitors. Support HERCULES through VGA cards. |
| MOVE | Move files to different locations on the hard disk without having to copy and delete manually. Supports multiple hard drives including network drives. |
| NAMEDIR | Renames directories without emptying or modifying their contents. |
| **NO** | **File exclusion utility. Lets you run a command on all files except specified groups.** |
| **ORDERFRM** | **Order form.** |
| PATHFIND | Allows you to search the file path for any files. Will display the files that it finds in order of appearance within the path. This is handy when you are trying to locate a similar program or file that could exist in multiple places in your path statement. |
| PSPRINT | Allows you to print DOS ASCII Text files on a PostScript device without the usual conversion. |
| SCRNATTR | Sets the foreground, background, and border colors at the DOS prompt. |
| SPEEDRAM | Increases the speed of your system CPU by changing the refresh rate of your system's 8253 timer chip. SpeedRam should increase your CPU's performance from 2 percent to 10 percent. |
| TEXTSRCH | Searches for all occurrences of a given string inside of a specified text file. |
| TIDY | Scans drives for any .BAK and 0 length files and deletes them automatically. |
| TOUCH | A UNIX-like utility used to stamp a file's attributes with the current system date and time. |
| **TURBOBAT** | **Compiles batch files into .COM binary programs. This will greatly increase the speed of large batch files by allowing them to run in machine code rather than interpreted DOS commands.** |
| TURBOTXT.TXT | Text Filer Compiler. |
| USEMEM | Program allows you to use up memory for testing programs and applications with a minimal memory setting. |
| USEMOUSE | Gives any DOS application the ability to use the mouse. |
| VIDMODE | Set the current video mode from the DOS prompt. Allows switching to any of the supported text modes. |
| VOLLABEL | Allows you to view and modify the volume label on any disk. |
| **WARRANTY.TXT** | **Warranty information.** |

## ALARMCLK

**DOS Alarm Clock**
**Version 2.00**

### Introduction

ALARMCLK provides a simple alarm clock for MS-DOS machines. ALARMCLK is split into two programs. You may install as many alarms as you wish per session. ALARMCLK is the alarm clock TSR that loads into memory and keeps track of the alarms that you specify. Once ALARMCLK is loaded, use SETALARM to set as many alarms as you like.

### Command Format

```
ALARMCLK [/D] [/U] [/Inn]
SETALARM [time] [PM] [/D] [/U]
   [/Inn]
```

| | |
|---|---|
| *TIME* | Time in HH:mm format. 12-hour or 24-hour clocks are both accepted for the time format. |
| *PM* | ALARMCLK will assume AM with 12 hour format unless PM is defined. |
| */D* | Deactivates all alarms |
| */U* | Removes ALARMCLK.COM from memory |
| */Inn* | Specify a new ID number |

## ALLSUB

**Perform a Task in All Subdirectories**
**Version 2.05**

### Introduction

ALLSUB allows you to run any command or program in the current directory as well as in all subdirectories of the current directory. This is useful for deleting groups of files or performing tasks with programs that don't normally allow for subdirectories. As the command is performed in each subdirectory, ALLSUB will display the directory of execution.

### Command Format

```
ALLSUB [command]
```

*[command]* is any valid DOS command or program.

## CALENDAR

**Display monthly or yearly calendar**
**Version 1.33**

### Introduction

CALENDAR displays a calendar from DOS in one of two formats, either one month or 12 months of one year.

### Command Format

```
CALENDAR [mm] yyyy [/C#] [/L##]
   [/T##]
```

| | |
|---|---|
| *mm* | month [1..12] |
| *yyyy* | year [0..9999]. If 2 digits are used, the year will be 19yy. |
| *C#* | display in # columns [2..3(default)] |
| *L##* | shifts display left by ## chars [0(default)..80] |
| *T##* | shifts display down by ## lines [0(default)..40] |

**Example**

```
CALENDAR 66 /C2
```

will display a calendar in 2 columns for the year 1966

*Note:* If you want to pause the display of a full year, press [CTRL]S to pause the screen and then any key to continue the display.

# CLEANUP

**Disk Drive Cleanup & Space Recovery**
**Version 2.06**

## Introduction

With the ever increasing size and lowering cost of disk space, it isn't uncommon to find PCs with an average of 80MB. Having larger disk drives has also made managing them become more complex. It is no longer feasible to remember the name and location and purpose of every file on your system. Often files and programs will be installed on a temporary basis only to be forgotten and either outdated or left behind. You can quickly devote as much as 25 percent of your hard disk to unused or duplicate files. Cleanup provides a comprehensive way of dealing with this problem by quickly isolating unused and duplicate files and removing them, freeing up valuable hard disk space. Cleanup displays the name of each file as it is deleted and a summary showing the total amount of disk space that has been recovered.

*Warning:* The main function of Cleanup is to recover lost disk space by removing duplicate and unused files. You have total control over which files will and will not be deleted, but with large hard disks it is easy to overlook single file names and accidently delete a desired file. It is suggested that you back up your disk before running Cleanup. This will ensure that you have a means of recovering files that you delete by accident. The other alternative is to have a copy of Norton Utilities UnErase or a similar type product.

## Command Format

```
CLEANUP [/L50]
```

*/L50* Force into 43/50 Line mode on EGA/VGA monitors

## Using Cleanup

Cleanup is a menu-driven program. After initializing the program, you are prompted with a list of disk drives in your system. In this list you are allowed to toggle the search amongst any installed drives in your system. Drives that are allowed but not attached are "grayed" out, and you cannot select these drives. When you select a drive, a check mark will note when a drive is selected. You may select any combination of all available drives. All of the drives selected will be treated as one large volume across which all duplicates and file marking will take place. This is particularly handy if you have more than one hard drive on which you keep files, and you might end up with duplicate files across the drives. Once you have selected the drives to scan, press the [F10] key to start scanning.

After all of the directories have been scanned, you will be located at the main menu of Cleanup. In the upper right hand corner of the screen, is the Current Status window. This window will always display information about the program's status. The number of directories, files, and duplicate files is calculated after selecting the drives and will not change until you exit the program and run again. The number of tagged files and bytes will show you the current statistics for the files that have been tagged for deletion.

These numbers will be updated as you tag or untag files for deletion.

There are two methods for tagging files for deletion. The first method is to use the Auto Tag and Auto Untag menus. These menus are identical in options, and perform exactly the opposite tasks. The Auto function allows you to tag groups of files all at one time. Once tagged, you may then go in and manually untag individual files using the manual tag option.

**The following are the options available from the Auto Tag and Auto Untag menus:**

| | |
|---|---|
| ALL | All allows you to tag/untag all of the files scanned. This is generally only useful for untagging files. Be careful as if you use this to tag files, you will tag all of the files on the selected drives for deletion. |
| SMALLER DUPLICATES | This will look at the list of duplicate files and whenever there is a difference in size, the smaller file will be marked for duplication. |
| OLDER DUPLICATES | This is quite handy and will compare all of the duplicate files and mark the oldest files. This is useful when you wish to keep only the most recent version of a file on your system. |
| BACKUP FILES | This option will mark all files ending in .BAK. This was the original function of the TIDY program from which Cleanup evolved. Many programs and editors create a backup of a file when updating it. After a while these backups can clutter up your system. |
| TEMP FILES | There are many programs that make use of temporary files while processing work. Sometimes, if these programs are interrupted or crash, the temporary files will be left on your drive. Most software uses the convention of naming temporary files with a .$$$ extension. This option will mark all files ending in .$$$. |
| EXT xxx | This allows you to tag a group of files with a particular extension that you know should be removed. With this option you can specify any group of files by giving the extension name. All files ending in the extension that you provide will be tagged. Be careful that you wish to delete all files ending in the extension before running the Erase option. |
| ZERO LENGTH | This was the other option available on the TIDY program. Another type of temporary file that can often be created is one that was opened but never properly closed. This will often result in a file entry in your directory with a file size of 0 bytes. The zero length is misleading. Every file entry requires at least one cluster; thus ,the minimum disk space used by a 0 byte file is the cluster size (512 bytes or more). Having a lot of 0 byte files can actually take away space from your drive. Choosing this option will mark all files with a size of 0 bytes for deletion. |
| LENGTH n | Like the Zero length files above, there may be instances where you create a lot of small files that you wish to tag for deletion. This option allows you to input a specific file size. All files matching this file size will be tagged for deletion. |

The MANUAL TAG option from the Main Menu allows you to view the list of files and to manually tag or untag files. When you first enter this screen, you will be presented with the list of duplicate files. To view all of the files in the system, press the [F6] key. Then to switch back to the list of only duplicate files, again press the [F6] key. Tagged files will be marked with a check mark next to the name, and will appear in red on color monitors. To view the list, use the up and down arrow keys. To tag a particular file, press the [ENTER] key. Pressing it a second time will untag the file. To untag a tagged file, highlight the file and press the [ENTER] key. Once again, the [ENTER] key will continue to toggle the tag on and off. When you are done viewing the manual tag list, press the [F10] key.

Once you are happy with the list of tagged files, choose the ERASE TAGGED option from the main menu. This will erase all files that were previously tagged. There is no UNDO feature on this program, so once they are deleted you will have to use an unerase program such as Norton Utilities Quick Unerase to recover any erased files. If you have accidentally deleted a file, do not use the disk drive until the file has been unerased. Changing data anywhere on the disk could result in permanently losing the information in the file.

***Warning:*** There have been some reports of problems when using Cleanup in a machine that is running both FASTOPEN and SMARTDRIVE. The best solution is to remove the usage of FASTOPEN as it can cause sorting problems with the file list on large disk drives.

# DISKINFO

**Disk Drive Information Display**
**Version 1.4**

## Introduction

DISKINFO is a utility that will read the boot sector of a disk and then make a DOS call to find out the information that DOS has for that disk. It will then list out the information in a side-by-side format so that you may compare the reports from DOS and the boot record. Most items reported are self-explanatory. Every disk is divided up into tracks, sectors, and clusters.

| | |
|---|---|
| *track* | is a single ring around a disk. |
| *sector* | a pie-shaped wedge of a track. |
| *cluster* | is the smallest unit to which DOS can interact with the disk drive, and it is a group of Clusters. |
| *System ID* | is the 8 byte record that is put on the disk when it is formatted. |
| *Media Descriptor* | is a byte value that describes what type of media the disk is. |

## Command Format

```
DISKINFO [d:][/?][/NOH]
```

| | |
|---|---|
| *d:* | is the drive to view; if no drive is specified, information on the current drive will be displayed. |
| */?* | Displays help information. |
| */NOH* | Suppress Header Display. |

# NO

**DOS File Exclusion Utility**
**Version 2.00**

## Introduction

NO is a utility that gives many DOS commands and external utilities the ability to exclude certain groups of files from the effects of the program called.

## Command Format

```
NO [filespec] [command parameters]
```

| | |
|---|---|
| *filespec* | is any valid DOS file specification including wildcards |
| *command* | is any DOS command, or external program |
| *parameters* | normal parameters passed to the command or program |

## Notes

NO hides files in the current directory from most programs and utilities by putting the

HIDDEN file attribute on. It then calls the command or program with the specified parameters. Once the command is complete, NO will remove the HIDDEN file attribute from the specified files.

NO is very useful when using commands such as DEL *.* when you want the action to take effect on all but a certain group of files.

NO is recursive, which means you can stack groups of excluded files together by stacking NO commands.

## Examples

```
NO *.EXE DIR
```

will display all files in the current directory with the exception of files ending in .EXE.

```
NO FILE.* DEL *.*
```

will delete all files in the current directory except those beginning named FILE with any extension.

```
NO *.EXE NO *.COM DEL *.*
```

will delete all files in the current directory except .EXE and .COM files.

# TURBOBAT

### Turbo Batch File Compiler

## Introduction

TurboBAT allows you to compile standard DOS batch files into .COM binary programs. This will greatly enhance the speed of large batch files by allowing them to run in native code rather than interpreted DOS commands. The resulting .COM files may call nested batch files without losing the parent since the parent calling program is now no longer a batch file.

## Command Format

```
TURBOBAT [/1][/4][/A][/Dname][/I]
  [/L][/T][/X] file[.ext]
```

| | |
|---|---|
| */1* | Insert single-stepping code. This creates a version of the output that is useful when debugging a batch file. When the output .COM program is run, it will single-step through each line of the original program. |
| */4* | Supress 4DOS warning messages. Don't report any warning messages generated by the use of 4DOS-specific commands. |
| */A* | Supress all warning messages (same as /4/I/L/X). Don't display any warning messages, only error messages. |
| */Dname* | Include full symbolic tracing information. If NAME is specified, the information will be sent to the named file; otherwise is displayed via the standard error device. There must not be any spaces between /D and NAME. |
| */I* | Ignore nonstandard labels. DOS supports labels up to 8 characters long for use with the goto statement. Many people use the label as a comment field rather than a label, and this can cause problems when a label that is used in a non-standard fashion shows up during the compile stage of the batch file. DOS will normally just ignore this error. |
| */L* | Relaxed label-length checking. The normal label size recognized by DOS is 8 characters. In many cases, people use the : as a placeholder for the program without ever referencing the label. The problem is that if you have more than one long label name where the first 8 digits match another label, the file won't compile because it will look as though you have duplicate labels. This switch relaxes that limit. |
| */T* | Compile with line-number TRACE information. The line-number information is displayed via the standard error device. When running the output program |

compiled with this option on, you will get a display of each line number as the program is running. This is useful when trying to determine where a program is failing.

/X Suppress extension warning messages. Normally during compilation, TurboBAT will warn you about any lines that make use of commands that are supported by the compiler but are not supported by DOS. Using the /X will tell TurboBAT not to warn you about any BATCH files that will not function under normal DOS conditions.

/B- Use ANSI escape sequences instead of the BIOS. The default screen I/O is now handled by making calls to the video BIOS. This ensures that the compiled batch file will work on any machine with or without ANSI.SYS installed. The drawback to this is that BIOS calls are much slower than ANSI calls. If you know that the machine that will be running the compiled program and then using the /B- command will speed up the video calls.

## Notes

TurboBAT is a multipass batch file compiler. It takes interpreted DOS batch files and turns them into binary programs that can execute up to as much as 4 times faster than normal batch files. While supporting all of the normal DOS functions that can be called from within a batch file, TurboBAT also provides many additional features including support for many 4DOS/NDOS commands.

## Extended Batch Commands

TurboBAT supports many commands beyond those supported in normal batch files. Some of these commands are in support of J.P. Softwares 4DOS Batch Enhancements, and others are to replace small external DOS utilities.

## Internally Supported Commands

DOS Commands and batch file commands that are directly supported by TurboBAT and require no external programs are as follows:

%? "Internal" environment variable. Expands to the error level of the last executed program as a string.

Example: `SET LASTERROR=%?`

BEEP BEEP [frequency duration ...]

Sounds a tone of the specified frequency for the duration in clock ticks. The defaults are 440Hz (A below middle C) for 2 clock ticks (about ¼ second).

The list of tones and durations can be repeated on a single line. The table is identical to the 4DOS tone table.

***Note:*** Please see the file TURBOBAT.TXT for table omitted here.

BOOT `BOOT [WARM | COLD]`

Boots the computer. If no options are specified or if WARM is specified, a WARM boot will occur. If COLD is specified, the computer will go through the entire cold boot process. Under Desqview, a warm boot will only close the window.

CDD `CDD drive:path`

Changes subdirectories and drives. This command will change the current directory and drive to the one specified in [drive:path] and make the current DOS drive equal to the value in [drive]. Normally within DOS the CD command will change directories on any drive that you specify but will keep the current drive the same. CDD allows you to switch to that drive rather than stay on the current drive.

CHDIR/CD `CD path`

`CD drive:path`

Changes the current directory

| | |
|---|---|
| | as specified in path. If a drive is specified, changes the current directory of that drive to the directory specified in path. |
| CLS | `CLS [[bright] [blink] fg on bg]`<br>Clears the screen and optionally sets the screen color. When setting the colors, the syntax is identical to the COLOR command. |
| COLOR | `COLOR [bright] [blink] fg ON bg`<br>Uses ANSI escape sequences to set the screen color. Only the first three letters of each command word are actually significant. The following colors are allowed:<br>BLACK BLUE GREen RED<br>MAGenta CYAn YELlow WHIte<br>Example: COLOR BRIGHT WHITE ON BLUE |
| DELAY | `DELAY [seconds]`<br>Pauses for the specified interval in seconds. If no duration is given, it will pause for 1 second. |
| ECHO | `ECHO [message]`<br>If the /D switch was used at compile time, you can enable or disable command-echoing with ECHO OFF and ECHO ON. ECHO without any text will indicate if the /D switch was used when the file was compiled, not the current status of command echoing. To display a blank line, use "ECHO:" or "ECHO.". |
| ECHOS | `ECHOS [message]`<br>Displays the message without appending a cr/lf to the end of the message like ECHO does. (Used to be ECHONLN but was updated to match the 4DOS command.) |
| ELSE | If the IFF condition was false, the lines from ELSE to ENDIFF will be executed. See Also IFF. |
| ELSEIFF | `ELSEIFF condition [THEN]`<br>Similar structure to the IFF command. If the previous IFF was false and the current condition is true, the code following will be executed to the next ELSE, ELSEIFF, or ENDIFF statement. Each use of ELSEIFF counts as a nesting level. The compiler actually converts this command into logical ELSE^IFF^ENDIFF statements. See Also IFF. |
| ENDIFF | Terminates an IFF statement. See Also IFF. |
| EXIT | EXIT is used when at the DOS level if you are running within a shelled version of the command interpreter and you wish to exit the shelled session. The EXIT command is ignored by TurboBAT. See also QUIT. |
| FOR | `FOR %%var IN (file list) [DO] command`<br>Allows more than a single letter as the loop variable. Can be nested. The loop variable can be tested from other lines by treating it as a normal environment variable. The word "DO" is required by DOS but is optional under 4DOS and Turbobat. |
| GOTO | `GOTO label`<br>Will jump control to the label. Labels are marked as a line beginning with a :. |
| GOSUB | `GOSUB label`<br>Calls a label in the current batch file as a subroutine. Must have a matching RETURN statement. The end of the batch file does *not* act as a RETURN. |
| IF | `IF [NOT] first (== \| EQ \| NE \| LT \| LE \| GT \| GE) second command`<br>`IF [NOT] ERRORLEVEL [== \| EQ \| NE \| LT \|LE \| GT \| GE] value command`<br>`IF [NOT] ENVFREE [== \| EQ \| NE \| LT \|LE \| GT \| GE] value command`<br>`IF [NOT] EXIST filename command.`<br>`IF [NOT] ISDIR path command` |

| | |
|---|---|
| *EQ* | (equal), also '==' and '=' |
| *NE* | (not equal), |
| *LT* | (less than), |
| *LE* | (less or equal), |
| *GT* | (greater than), |
| *GE* | (greater or equal). |

If no compare command is used for ERRORLEVEL or ENVFREE, the default of GE will be used.

EXIST tests for the existence of a file.

```
IF EXIST my.cfg GOTO
  continue
ECHO Configuration file
  is missing
QUIT 99
:continue
```

ISDIR tests if the given path specification is an existing subdirectory.

```
IF NOT ISDIR \work MD
  \work
```

ENVFREE tests the available environment space.

IF ENVFREE LT 123 ECHO Insufficient environment space

**IFF**

```
IFF condition [THEN]
```

If the condition is true, the lines up to ELSE, ELSEIFF, or ENDIFF (whichever appears first) will be executed. IFF statements may be nested up to 15 levels deep. See also IF.

```
IFF %? EQ 0 THEN
GOTO CONTINUE
ELSEIFF %? EQ 99
THEN GOTO CRITICAL
ELSE
GOTO ERROR
ENDIFF
```

**INKEY**

```
INKEY [/Wn] [message]
%%var
```

Gets a single keystroke and places it in the environment. %%var is the environment variable that will be assigned the keystroke. If /W is specified, the command will wait up to that many seconds, (e.g., /W10). It will also optionally display a message. If a wait of zero is used (/W0), the command will return immediately if no keystrokes are pending. The wait period can be specified in the environment, but the /W must be present on the line at compile time, for example:

```
inkey /w%delay Press
  any key: %%key
```

Displayable keystrokes (characters above the space) are stored as their corresponding character. Keystrokes that do not have a corresponding character are displayed as a number, (e.g., the carriage return as "13"). Extended keystrokes such as F1 are stored with a preceding '@', (@59 for F1).

**INPUT**

```
INPUT [/Wn] [message]
  %%var
```

Gets a string and places it in the environment. %%var is the environment variable that will be assigned the string. The string is terminated by a carriage return or end of file mark (^Z). Extended keystrokes are ignored. /Wn is the optional maximum time in seconds to wait for the first keystroke. Once a keystroke is entered, this command will not time out.

**MKDIR**

```
MKDIR [pathname]
```

**MD**

```
MD [pathname]
```

Will create a new directory of PATHNAME.

**PATH** Displays or Sets the Path.

**PAUSE**

```
PAUSE [message]
```

Displays an optional message. If no message is specified, the default of "Press a key when ready..." is used. In addition, a CR/LF is not output until after the key is pressed.

**PROMPT** Displays or Sets the DOS Prompt.

**QUIT**

```
QUIT [exitcode]
```

Unconditionally halts the compiled batch file. If no exitcode is specified, zero will be used. The exit code can refer to an environment variable. If the variable is not a number or does not exist, zero will be used. Examples:

```
SET var=1
QUIT %var
```

Exits with 1.

```
SET var=test
QUIT %var
```

Exits with 0 because %var is not a number.

**REM** REM is the standard DOS Remark field. REM lines are ignored by the TurboBAT Compiler.

**REPEAT/ UNTIL** Similar to the Pascal structure. May be nested up to 15 deep. Redirection is not allowed on a line containing REPEAT or UNTIL.

```
REPEAT
ECHO %1
SHIFT
UNTIL "%1" EQ ""
```

**RETURN** Return from a subroutine call. If no subroutines are pending, it will terminate the batch file with an exit code of 255.

**RD/RMDIR** Remove Directory. This will remove the directory specified as long as there are no files in it.

**SCREEN**

```
SCREEN row column
[message]
```

Positions the cursor using ANSI sequences and displays an (optional) message. The cursor coordinates are 0 based, the normal range being 0–24 rows and 0–79 columns.

**SCRPUT**

```
SCRPUT row col [bright]
  [blink] fg ON bg
  message
```

Writes the message at the specified screen coordinates using the specified colors. The cursor position is restored to the position before the command. ANSI escape sequences are used throughout.

**SET** Alone, SET will display the current environment variables, or followed by an environment variable, SET will set the environment variable to the given value.

**SHIFT** Same as standard DOS. I will be expanding this command for 4DOS compatibility to accept a shift factor, with support for a negative number which will get back previously shifted parameters.

**SINGLESTEP**

```
SINGLESTEP ON | OFF
```

Enables or disables single-stepping. The /1 compiler switch must be used for this command to have any effect. The options are resolved at compile time so you cannot use an environment variable or command-line parameter to set the status.

**TEXT/ ENDTEXT** The text that follows, up to ENDIFF, will be displayed as if each line was preceded by an ECHO. Output redirection is more efficient using this command instead of individually redirected ECHO commands since the file will only be opened and closed once, while the file will be opened and closed for each ECHO command.

```
TEXT
This could be a menu or
it could be help
information
ENDTEXT

TEXT >dummy
This text will be sent to
the redirection file.
Up to, but not including
the ENDTEXT command.
ENDTEXT
```

**UNSET** Deletes the environment variable from the master environment.

```
UNSET RESULT
```

**WHILE/ WEND** Similar to the Pascal structure. May be nested up to 15 deep. Redirection is not allowed on a line containing WHILE or WEND.

```
WHILE "%1" NE ""
   ECHO %1
   SHIFT
WEND
```

## Commands Handled by Calling the Comspec

These commands are supported by calling COMMAND.COM:

CALL, DIR, DEL, REN, VER, VOL, CTTY, CHCP, TYPE, COPY, DATE, TIME, ERASE, BREAK, RENAME, DELETE, VERIFY, COMMAND

These commands will only work if 4DOS is the active COMSPEC:

FREE, DRAWBOX, DRAWHLINE, DRAWVLINE, MEMORY COMPILER ERROR MESSAGES

TurboBat contains a library of internal error messages and will display the error number and message when an error or warning is found. Files with just warnings will compile, files with errors will not compile.

***Note:*** Please see the file TURBOBAT.TXT for a section on warnings and error messages omitted here.

# TURBOTXT

**Text File Compiler**
**Version 1.11**

## Introduction

TurboTxt will convert a standard text file into a self-displaying COM program, containing both text and the necessary code to display the text.

## Command Format

```
TURBOTXT [/Axx] [/Fxx] source
   [destination]
```

| | |
|---|---|
| */Axx* | Sets the video attributes for the text being displayed. xx is the attribute value. Use the chart below to calculate the attribute value you wish to use. |
| */Fxx* | Sets the video attributes for the help line at the bottom of the screen being displayed. xx is the attribute value. Use the chart below to calculate the attribute value you wish to use. |
| *source* | The source text file to be compiled. Can be any standard DOS text file up to 63k bytes. |
| *destination* | Can be any valid DOS filename. If no destination is entered, the file created will be called README.COM. |

## Colors

TurboTXT allows you to choose the video attributes that will be used to display the text in the compiled README.COM program. The number used to set the attribute is calculated by choosing the background color and then the nearground color. The formula to use to get this number is:

A = BACKGROUND COLOR x 16
B = NEARGROUND COLOR
xx = A + B;

| *BACKGROUND COLORS* | | *NEARGROUND COLORS* | |
|---|---|---|---|
| BLACK | 0 | BLACK | 0 |
| BLUE | 1 | BLUE | 1 |
| GREEN | 2 | GREEN | 2 |
| CYAN | 3 | CYAN | 3 |
| RED | 4 | RED | 4 |
| MAGENTA | 5 | MAGENTA | 5 |
| BROWN | 6 | BROWN | 6 |
| LIGHTGRAY | 7 | LIGHTGRAY | 7 |
| | | DARKGRAY | 8 |
| | | LIGHTBLUE | 9 |
| | | LIGHTGREEN | 10 |
| | | LIGHTCYAN | 11 |
| | | LIGHTRED | 12 |
| | | LIGHTMAGENTA | 13 |
| | | YELLOW | 14 |
| | | WHITE | 15 |

So to display LIGHTRED text on a BLUE background you would use the following calculation:

A = 1 x 16 = 16
B = 12
xx = 16 + 12 = 28

```
TURBOTXT /A28 README.TXT
```

## Notes

The code overhead is slightly more than 1k bytes. When this program is executed, the displayed text can be scrolled with the cursor keys. Direct screen writing is used so the display is extremely fast. Snow suppression will occur if a CGA monitor is detected. The text file must be less than 63K in size, and any lines exceeding the width of the screen will be truncated. It is possible for the resultant COM program to be smaller than the original text file. Most text files use a two-byte sequence of a carriage return and line feed (0Ch and 0Ah) to terminate each line. When the text is converted, the two bytes are replaced by a single null byte (00h). Needless to say, the larger the text file, the better the chance of a size savings.

If the output file is not specified, README.COM will be used. If the extension is omitted from the output filename, COM will be used.

The following keys are available when the created program is executed:

| | |
|---|---|
| Up Arrow | Scroll up one line |
| Down Arrow | Scroll down one line |
| Page Up | Scroll up one page |
| Page Down | Scroll down one page |
| Left Arrow | Scroll Left one character |
| Right Arrow | Scroll Right one character |
| Home | Go to start of text |
| End | Go to end of text |
| ESC | Exit program |
| Alt-X | Exit program |

The .COM program produced automatically supports screen sizes other than 80x25. As long as the proper screen values are in the BIOS data area and the screen width is at least 80 columns and no greater than 255 columns, it will work with the detected mode automatically.

**Example**

```
TurboTxt help.txt help
```

The text in HELP.TXT is used to create HELP.COM.

## ExtraDOS Toolbox Professional Registration Form

Name: ______________________
Company: ______________________
Address: ______________________
City: ______________________
State: __________ Country: ______________
Zip: __________ Phone: ______________
Fax orders: Country Code and number: __________

**Remit to:**
Foley Hi-Tech Systems
185 Berry Street
San Francisco, CA 94107

VOICE:(415) 882-1730, Mon.–Fri., 9:00 a.m. - 5:00 p.m. Pacific.
FAX: (415) 882-1733
BBS: (415) 882-1735, 8N1, 12-9600 BPS, V22.bis, V.32.bis, V.42.bis, HST 14.4K
CIS: 70262,1642

We offer discounts on quantity orders, multiple-user licenses, and dealer pricing. Please call for details. We accept Visa, MasterCard, company or personal checks. You can order the entire toolbox or individual utilities by listing them.

FHTS, please send:
ExtraDOS Toolbox License Registration w/ manual, (all 52 FHTS Shareware Utilities) each ........ $59.95 $ ______
Copies of ____________ Single Utility each ........ $19.00 $ ______
Copies of ____________ Single Utility each ........ $19.00 $ ______
Copies of ____________ Single Utility each ........ $19.00 $ ______
CA residents, add sales tax ........ 8.25% $ ______
Shipping and Handling
U.S.: ........ $2.95 $ ______
Canada ........ $5.00 $ ______
Other International Locations: .... $9.00 $ ______
TOTAL (U.S. Funds drawn on U.S. Bank) .. $ ______

Specify Disk Size: ☐ 5.25" 360K ☐ 3.5" 720K
Payment by:
☐ Check (Mail only) ☐ MasterCard ☐ Visa
Call for Corporate P.O. Approval
Credit Card #: ______________________
Exp. Date: ______________________
Authorized Signature: ______________________

Prices and terms subject to change without notice.

# FANSI-CONSOLE

**Fast, Powerful ANSI.SYS Replacement**
**Version 3.00**
**Copyright © 1984–1991, Hersey Micro Consulting, Inc.**

## What Is FANSI-CONSOLE?

Technically, FANSI-CONSOLE is a complete replacement for the IBM or compatible MS(PC)-DOS console driver as well as the screen and keyboard parts of the ROM BIOS. More importantly to you, it has many features that make the use of nearly every IBM-PC application program more convenient. It does so many things, we also call it "The Integrated Console Utility."

## What Does FANSI-CONSOLE Do for Me?

- Speeds up all screen writing done through MS(PC)-DOS or ROM BIOS calls from 1.2 to 3.0+ times, or from 1.4 to 8.0+ times with optional quick mode hardware scrolling! Includes the screen writing of ALL the standard MS(PC)-DOS commands such as TYPE, DIR, ECHO, and MORE. Speedup amount depends on computer, display adapter type, and application program.
- Provides a scroll recall facility which allows reviewing lines which have scrolled off the screen which you would otherwise lose forever! You can save up to about 800 screenfuls using Expanded Memory managed by any other device driver which meets the Lotus/Intel/Microsoft Specification. Or you can save fewer lines using some regular MS(PC)-DOS memory. You may save the scrolled line buffer to a file or cut & paste into nearly any program, such as word processors. If not automatically saved, you can take a snapshot of a screen. You can save the lines in full color, or you can conserve memory and just save the characters.
- Adds zip to your cursor keys and other keys, up to 5 times typically!
- Adds a much larger typeahead buffer (255 characters)!
- Optional automatic single or dual screen display dimming (both screens in a two-screen computer) after specifiable period of inactivity! Prevents burning patterns into your screen phosphors.
- Frees your eyes from CGA scroll blinking!
- Provides many more ANSI X3.64 standard escape sequences than IBM's ANSI.SYS program! Space prevents us from listing them all here; ask about any you are specifically interested in. It supports nearly all the VT100 series escape sequences and many more! Has at least 50 extra private RM/SM mode settings and at least 15 extra private multivalue mode settings, giving full control over a multitude of screen and keyboard features.
- A DEC VT100/VT52 emulation toggle which provides VT100 emulation when used with our MODEM86 or many other communication programs! Programming support for separately scrolled windows via ANSI X3.64 escape sequences (SSR)! You can use the escape sequences with any programming language. Your program thinks it is writing to an ANSI or VT100 style terminal!
- Optional ANSI X3.64 sequences processed during BIOS TTY calls!
- Optional one finger pausing!
- Optional controls to change the behavior of shifting and locking keys! Rearrange keys like switching Left-Shift and BackSlash! Optional Dvorak keyboard arrangement!
- Keyboard induced debugging breakpoints for assembly language programmers!
- Keyboard macros just like those in ANSI.SYS, with option to have them expanded all the time instead of just for DOS calls! More memory for ANSI-compatible keyboard macro strings than standard IBM-PC ANSI.SYS device driver allows! You can specify how much memory to use.

- Shorten bell time length!
- Add a key click!
- Extends IBM-PC screen and keyboard ROM BIOS to be IBM-PC AT compatible!
- Extends EGA ROM BIOS to be more VGA compatible!
- Control over colors used by many programs!
- Software support for 50 line CGA displays with interlace, or doubled scan lines with interlace for those screen adapters that have interlace hardware capability!
- Support for 43 line displays with Enhanced Graphics Adapters (EGA) and Enhanced Graphics Displays (EGD)!
- Software support for differing (larger) screen text dimensions for those compatible display adapters that have that hardware capability!
- So many other little features, they just don't fit here! Nearly every feature is optional and easily changeable using a menu-driven program!
- Easy installation, just like MS(PC)-DOS ANSI.SYS. Place one line in the CONFIG.SYS file and MS(PC)-DOS automatically loads FANSI-CONSOLE every time you start MS(PC)-DOS.

## Is FANSI-CONSOLE Compatible with My Hardware and Software?

FANSI-CONSOLE is compatible with thousands of existing programs. It supports all IBM compatible modes of operation on most IBM-PC compatible computers and adapters, including AT, 386, 486, PS/2, EGAs, and VGAs. It may not yet support some extended functions unique to specific compatibles.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the FANSI-CONSOLE files in the directory \SECRETS\FANSI. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the FANSI-CONSOLE files to a directory that is included in the search path.

***Note:*** The documentation for FANSI-CONSOLE is quite lengthy, and therefore, it is not reprinted in its entirety here. Please see the on-line documentation for complete details. Of particular interest are Chapter 2, entitled "Starting Quickly" (especially Section 2.2 "How Do I Quickly Install FANSI-CONSOLE?"), and Chapter 3, entitled "Detailed Installation." Sections 2.1 and 2.3 are included here for your convenience.

## Do I Need to Read the Whole FANSI-CONSOLE User Manual?

NO! You do not need to read the whole manual from cover to cover to take advantage of FANSI-CONSOLE. You only need to know the information in this chapter to get started. If you want really brief instructions, just read the description of this chapter in the table of contents! It really tells you what to do! We know how much you hate to read the manual before trying out a program. Unfortunately, the nature of FANSI-CONSOLE prevents us from making it menu driven.

The manual has an extensive index at the end to help you when you run into problems finding what you want to know. It also has a glossary at the end to help you find the meaning of technical terms and abbreviations. Please use them, especially if you do not read the manual from cover to cover. For example, you can look up the keyword "problem" in the index to find solutions to many common problems. This includes those you may run into when you install FANSI-CONSOLE.

If you cannot find something where you first looked in the index, let us know! We think the index should be comprehensive. That minimizes the number of calls we get asking about things the manual fully explains but you cannot find.

The chapter entitled "Compatibilities and Limitations" describes how to solve compatibility problems with FANSI-CONSOLE. You should read that if you experience problems.

If you can not find what you want to know in the abbreviated user manual found on the distribution diskettes, then it is probably time to order a complete, printed User Manual. You can tell this when you find yourself looking for a page number mentioned in the index or table of contents which the abbreviated user manual does not include. The index and the table of contents are complete, but the rest of the abbreviated user manual is not! The abbreviated user manual is only complete enough to install FANSI-CONSOLE and test it for compatibility with your hardware and software. It should convince you that FANSI-CONSOLE really is fast and professional, and that you really need the complete printed manual!

Lastly, even though you can use FANSI-CONSOLE without reading the chapter on its distribution, please read it, too. It has important information about what we expect in return from you for your being able to use FANSI-CONSOLE.

## How Do I Use FANSI-CONSOLE?

Now that you have installed FANSI-CONSOLE, you just proceed as you always do and mostly ignore that you installed FANSI-CONSOLE. When the FANSI-CONSOLE banner appears on your screen when you start MS-DOS, many programs magically write to the screen faster without your further intervention.

As noted before, you may want to use the typeahead buffer more than before. We made the extended typeahead buffer a non-optional part of FANSI-CONSOLE since FANSI-CONSOLE only uses the extended buffer when the normal buffer fills! So you do not need to use any FANSI-CONSOLE options to extend the typeahead buffer. You may also want to take advantage of the new special keys described here.

This section describes some keys that you can use that give extra functionality that they did not give without FANSI-CONSOLE. We occasionally refer to the *DarkPlus* and *DarkMinus* keys. These are just the slightly darker colored plus and minus keys on the right-hand edge of the keyboard. We do this to distinguish them from the lighter colored *LightPlus* and *LightMinus* keys just to the left of the BackSpace key at the top of the keyboard.

### Using the Ctrl-F key

Press the Ctrl-F key to simply flush (empty) the typeahead buffer. FANSI-CONSOLE leaves the Ctrl-F in the buffer, but many programs ignore it or do something harmless. If you use an application that does something dangerous with the Ctrl-F key, you may want to try the Ctrl-S key instead.

### Using the Ctrl-S key

Press the Ctrl-S key to flush the typeahead buffer and, if the current program uses Ctrl-S as a pause key, cause the current program to pause. Most programs pay attention to this key and pause; otherwise, they probably ignore it. The standard IBM-PC console software processes the Ctrl-S slightly differently. It does not flush the typeahead buffer when you press a Ctrl-S. This means that the standard IBM-PC console software ignores the Ctrl-S character if you already typed some characters into the typeahead buffer.

### Using the Ctrl-Num-Lock key

Press the Ctrl-Num-Lock key to cause the display to pause without flushing the typeahead buffer. With FANSI-CONSOLE, this key combination works similarly to the way it works with the standard IBM-PC console software. One advantage, however, is that unlike the standard IBM-PC console software, FANSI-CONSOLE prevents the appearance of duplicate lines during a pause. Press any other different key combination to unlock the pause. Unless the second key combination is a Ctrl-Break or a Ctrl-C, FANSI-CONSOLE ignores it other than for clearing the pause. Any program that works with FANSI-CONSOLE pauses with Ctrl-Num-Lock. For other ways to pause, see the descriptions of Ctrl-S in the previous paragraph, of Pause in the next paragraph, and of the /L option in the chapter entitled "Detailed Installation."

### Using the Pause key

For enhanced keyboards, the Pause key always remains a one-finger pause key. To use the one-finger pause key, press the Pause key to cause the display to pause without flushing the typeahead buffer. The Pause combination works differently than the Ctrl-Num-Lock combination. Unlike Ctrl-Num-Lock, Pause acts as a toggle. When you disable the scroll recall feature and you pause, you may use other keys to typeahead without affecting the pause. If you enabled the scroll recall feature, then you may do scroll recall when paused. Pressing Pause a second time clears the pause. Usually Pause is more convenient than Ctrl-Num-Lock mostly because it is a one-finger FANSI-CONSOLE pause key. Any program that works with FANSI-CONSOLE pauses with Pause. For other ways to pause, see the descriptions of Ctrl-S and Ctrl-Num-Lock in the previous paragraphs in this section.

## Using the Ctrl-C or Ctrl-Break keys

Press the Ctrl-C key to cancel most programs. FANSI-CONSOLE treats it the same way as the Ctrl-Break key for programs using MS-DOS input. It treats both keys the same way that the standard IBM-PC console software treats the Ctrl-Break key. It flushes the typeahead buffer before placing a Ctrl-C in it. The standard IBM-PC console software processes the Ctrl-C slightly differently. It does not flush the typeahead buffer when you press a Ctrl-C. This means that the standard IBM-PC console software ignores the standard MS-DOS Ctrl-C cancel character when you already typed some characters into the typeahead buffer. Pressing Ctrl-C does not cancel any program that Ctrl-Break does not cancel when using the standard IBM-PC console software.

## Using the Ctrl-Alt-Enter key

Press the Ctrl-Alt-Enter key to exit from many programs with infinite loops. It brings you back to the MS-DOS command prompt. If you used the MS-DOS CONFIG.SYS STACKS= command, then using the Ctrl-Alt-Enter key leaves MS-DOS considering one of these stacks permanently in use. MS-DOS remains OK when you do this once, but not if you do this repeatedly. You run out of stacks one at a time, and MS-DOS eventually fails. If you do not use the MS-DOS CONFIG.SYS STACKS= command, then using the Ctrl-Alt-Enter key does not cause this problem. If you press the Ctrl-Alt-Enter key when in the middle of an MS-DOS or other system software call, it may leave MS-DOS in an unstable state. So the best use consists of neatly saving your RAM-disk or other work files before restarting MS-DOS. Like all others, this key combination only works when keyboard interrupts remain enabled!

## What to do when key combinations do not work

Some computers have some special non-IBM standard keyboard functions when you do not install FANSI-CONSOLE. The key combinations for these functions frequently include Ctrl-Alt. This includes key combinations for such things as: configuration setup, CPU clock speed change, and screen intensity controls. For example, some computers change hardware clock speeds when you press a certain key combination, such as Ctrl-Alt-Backslash, or Ctrl-Alt-DarkPlus and Ctrl-Alt-DarkMinus. When you install FANSI-CONSOLE, these key combinations do not work the same way, unless FANSI-CONSOLE itself supports that function.

Instead, when you install FANSI-CONSOLE, you must press the Ctrl-Alt-Grave key combination once before each such key combination you want to use. The Grave character is the backward-pointing accent. You use the same key as for the Tilde (the squiggly line!) character. Make sure you press the Grave key last and release it before the Ctrl and Alt keys.

Pressing the Ctrl-Alt-Grave key combination tells FANSI-CONSOLE that you wish to use the original BIOS keyboard interrupt processor. So remember, Ctrl-Alt-Grave "brings the original keyboard BIOS back from the grave." It only affects the next key combination. It does not remain permanently in effect. However, you can do that too, if you do not mind losing most FANSI-CONSOLE keyboard features. See the description of the FANSI-ORIGKEY mode in the chapter entitled "Changing options at run-time." Note that the abbreviated user manual found on the diskettes does not include this chapter, but the complete printed user manual does.

If the special function associated with the key combination when you do not install FANSI-CONSOLE affects the keyboard or the screen processing, it usually does not work with FANSI-CONSOLE even if you press the Ctrl-Alt-Grave key combination first. This is because most such functions require continued support after completing the key combination function, and nothing can give that support because FANSI-CONSOLE has replaced the keyboard and screen handling. However, in these cases FANSI-CONSOLE usually has a similar function which you can do using a different key or some type of control sequence.

The Ctrl-Alt-Grave key combination is NOT a way to momentarily turn off keyboard macros! Although it happens to work for macros expanded at the BIOS level, FANSI-CONSOLE still expands DOS-level macros for any keycodes placed into the typeahead buffer with the help of the Ctrl-Alt-Grave key combination.

Unless FANSI-CONSOLE has the same control functions, no way exists to use macros or the FANSI-CONSOLE SEND command to duplicate the functions of those key combinations which you must precede with Ctrl-Alt-Grave. The original BIOS keyboard handler must process the scan codes directly from the hardware registers for those key combinations. No way exists for any program such as FANSI-CONSOLE to put things into these hardware registers for other programs such as the original BIOS keyboard handler. FANSI-CONSOLE CANNOT duplicate these functions this way. Unfortunately, you CANNOT make ANY shortcut way to do them without knowing how to do the functions and writing your own program to them. Specifically, you CANNOT use FANSI-CONSOLE keyboard macros or keyboard description files to do this. Instead, you ABSOLUTELY MUST press Ctrl-Alt-Grave and then really press those key combinations to do the functions.

## Special Key Table

In summary, FANSI-CONSOLE defines the following keys. It treats them differently than the standard IBM BIOS. These keys function this way within nearly all programs. This is a partial list. It just has the keys already described more completely in this chapter. The chapter entitled "Keyboard Arrangement" describes more special FANSI-CONSOLE keys and also has a more complete list at the end. Note that the abbreviated user manual found on the diskettes does not include this chapter, but the complete printed user manual does.

| | |
|---|---|
| Ctrl-Alt-Enter | Exits the currently running MS-DOS program. |
| Ctrl-Alt-Grave | Passes the next key combination to original BIOS keyboard interrupt handler. |
| Ctrl-Break | Call Ctrl-Break interrupt. Usually exits current program or stops process. |
| Ctrl-C | Flushes the typeahead buffer and adds Ctrl-C. Similar to normal Ctrl-Break. |
| Ctrl-F | Flushes the typeahead buffer and adds Ctrl-F. |
| Ctrl-Num-Lock | Pause without flushing typeahead buffer. Must press another, different key combination to unlock! |
| Ctrl-Prtsc | Converted to Ctrl-P. The MS-DOS command editor treats this as console printer echo toggle. |
| Ctrl-P | FANSI-CONSOLE does not change the way this key works. The MS-DOS command editor treats this as printer echo toggle. |
| Ctrl-S | Flushes the typeahead buffer and adds Ctrl-S. Many programs treat this as a pause. |
| Pause | Causes the display to pause without flushing the typeahead buffer. Must press the same key to unlock! |
| Others | If your computer has special non-IBM compatible key sequences, refer to the description of Ctrl-Alt-Grave. |

## Testing for FANSI-CONSOLE

You may want to write programs with capabilities which depend on having FANSI-CONSOLE installed. Any program may test whether you installed FANSI-CONSOLE. FANSI-CONSOLE has another device name besides CON:. You may also open it as FCON:. So, to allow your FANSI-CONSOLE program to test whether you installed FANSI-CONSOLE, simply make the program attempt to open the device FCON: for reading. If the open attempt succeeds, then you installed FANSI-CONSOLE. If the open FANSI-CONSOLE attempt fails, then you did not install FANSI-CONSOLE. If you did not install FANSI-CONSOLE, you may want your program to print an error message, or just use a slightly different mechanism for screen writing.

You can make a batch file do different things depending on whether you installed FANSI-CONSOLE. Just use the following example as an outline:

```
IF EXIST FCON ECHO FC HERE!
IF NOT EXIST FCON ECHO *NO* FC
   HERE!
```

MS-DOS only checks the first part of a filename — the characters before the extension — when it determines whether the filename is a device name. It ignores the filename extension when it does this. Thus, when you install FANSI-CONSOLE, you cannot read or write files with names like FCON.???. This includes directories! If you try, you really use FCON: (FANSI-CONSOLE) instead. If you write to FCON:, then FANSI-CONSOLE writes data to the console. If you read from FCON:, then FANSI-CONSOLE reads data from the keyboard.

For more information about detecting FANSI-CONSOLE, see the section entitled "How does my program detect whether I installed FANSI-CONSOLE?" in the chapter entitled "Programming Background" in the FANSI-CONSOLE Technical Manual.

Usually, if you uninstall FANSI-CONSOLE, you should install the standard MS-DOS ANSI.SYS console device driver instead. If you uninstall FANSI-CONSOLE, but you use programs, batch files, or a prompt that uses ANSI X3.64 control sequences codes, you need to replace the "DEVICE=FCONSOLE.DEV" line in the CONFIG.SYS file with a line containing "DEVICE=ANSI.SYS".

## FANSI-CONSOLE Registration Form

Name: ____________________

Company: ____________________

Address: ____________________

City: ____________________

State: __________ Country: ____________________

Zip: __________ Phone: ____________________

**Remit to:**

Hersey Micro Consulting, Inc.
P.O. Box 8276
Ann Arbor, Michigan 48107-8276
(313) 994-3259

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Specify Operating System: ☐ DOS ☐ OS/2

Payment by:

☐ Check ☐ Credit Card

Make checks payable to:
Hersey Micro Consulting, Inc.

Authorized Signature: ____________________

Card number: ____________________

Expires: ____________________

FPN FANSI-CONSOLE Complete Package New
(includes both software & both manuals)
each .................................... $119.95 $______

FMN FANSI-CONSOLE User Manual New
(includes software & user manual)
each .................................... $75.00 $______

FDN FANSI-CONSOLE Diskettes Only New
(includes software only)
each .................................... $49.95 $______

FSN FANSI-CONSOLE Master Serializer New
(includes software master & serializer)
each .................................... $44.95 $______

Shipping (see below) .................... $______

Michigan residents add sales tax ........ 4% $______

TOTAL ................................... $______

We accept Visa, Master Card, and American Express with no surcharge. Add shipping and handling (for 1 copy): UPS Ground to continental USA $4, UPS 2nd Day to continental USA $6, UPS 2nd Day Alaska/Hawaii $8, UPS Next Day $15, Fed Ex Priority 1 $30, Fed Ex Std Air $15.

# FormatMaster

**Version 4.6**
**A Menu Driven Floppy Disk Formatter**
**Copyright © 1987–92 by New-Ware**

## Getting Up And Running

Operation of FormatMaster is relatively simple and straightforward. Almost all operations are accomplished through selection of a menu item, either via the cursor bar and hitting ENTER or by pressing the key of the first letter of a menu item. However, some initialization steps should be taken before attempting to format any disks.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the FormatMaster files in the directory \SECRETS\FORMASTR. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the FormatMaster files to a directory that is included in the search path.

## Program Initialization

Here are the two things you should do to get the program to a fully functional state.

1. Select SET DRIVE TABLE from the main menu. FM will normally detect and use the correct parameters for the type of drives in your system, but if the actual drive information differs, use the drive table menu to make the appropriate changes.
2. Make sure that the DOS utility SYS.COM is in a directory that is listed in your path string and that this directory is on the system boot drive if you intend to transfer system files to a newly formatted diskette. See the file PATHINFO.TXT for more information on the DOS PATH command.

## FormatMaster Operation

FormatMaster is menu driven and therefore quite easy to use. The main menu contains the following selections:

```
Simultaneous Format/Verify
Format/Verify via DOS
Format/Verify via BIOS
Rapid Format
Only Verify
Define Drive Table
Enter DOS
Check Disk
Transfer System     OFF
Add Vol Label       OFF
cYclic Format       OFF
Beeps     ON
Verify    OFF
Quit to DOS
Print Registration
@ Configure FM
```

Menu items are selected either by moving the cursor bar to the desired item and pressing return (MOUSE LEFT BUTTON) or by pressing the key corresponding the highlighted character. Two of the items above are toggles, i.e., they are either on or off and each time you select one, it is toggled to the opposite setting. If TRANSFER SYSTEM ON is set, then FormatMaster will invoke SYS.COM to transfer system files to the freshly formatted disk. Make sure that COMMAND.COM is not a hidden file. If ADD VOLUME LABEL ON, then you will be prompted to enter a disk label prior to formatting any disks.

The main menu may be skipped by executing FormatMaster with the drive letter and a colon on the command line as follows:

```
FM a:
```

In this case, the main format routine window will appear and FormatMaster will be ready to format the drive whose letter was placed on the command line. It is suggested that some users might desire to configure different copies of FM.EXE in different ways and then load the desired copy with the drive letter on the command line to speed up operation and to minimize keystrokes.

All menu items are augmented by a help line that is written between the top two double lines of the screen.

## Formatting Options

The first three menu items on the main menu access different formatting techniques. The first of these is the SIMULATANEOUS FORMAT/VERIFY option. This will format and verify a diskette in the same manner that DOS accomplishes that feat. This option will be the slowest but safest method of formatting.

The second technique, FORMAT/VERIFY VIA DOS, is really only a variant of the above option. Here the disk is formatted using DOS first and then, if the Verify switch is set, a full verification is performed. Separating the two functions permits formatting without verifying. If you are using high quality diskettes, you can save a lot of formatting time by skipping the verification.

The third technique, FORMAT/VERIFY VIA BIOS, uses direct BIOS calls instead of DOS calls and is included for those who might still be using a DOS version less than 3.1 or those who have difficulty formatting add-on floppy drives via DOS.

Please note that if you are running a version of DOS that is earlier than 3.1, the first two menu items will be disabled.

## Formatting a Disk

Selection of one of the first three menu items brings up a menu of available system floppy drives. If this menu does not accurately reflect the actual drives in your system, then use the DEFINE DRIVE TABLE option on the main menu to properly configure FM for your system.

If the disk you have placed in the designated drive is already formatted, it may contain valuable data and FormatMaster will warn you of this condition and give you four options in a pop-up menu. Options on this menu are

OK — Continue
Abort
View
Check Disk
Quit To DOS

Use Abort to discontinue the formatting operation. The View option will display an abbreviated directory listing of the target drive so that you may decide if you really want to format the disk or not. The Continue option will proceed with formatting. Since the target disk is already formatted, you may desire to save some time by doing a Quick Format. The Quick Format option simply re-initializes the FAT and ROOT DIRECTORY areas of the diskette instead of doing a complete track-by-track format. Verification will proceed normally if that switch is ON. The View option will clear the screen and display the files on the target drive. The Check Disk option displays information similar to the DOS CHKDSK utility.

If the disk is not formatted, FormatMaster will check the capacity entry in the drive table and if the drive is a high capacity drive, you will be asked whether you desire to format a high capacity diskette. If the disk contains data, FormatMaster will automatically adjust to the proper format mode. After this query, FormatMaster will format the disk, updating the track counter as it formats each track. When the formatting is complete, if the TRANSFER SYSTEM toggle is ON, the system files will be transferred and if the ADD VOLUME LABEL toggle is on, the volume label entered from the main menu will be added to the target disk. A disk counter is incremented and displayed during cyclic formatting to help you keep track of the number of disks formatted.

If Cyclic formatting is not in progress, a YOUR CHOICE submenu is presented at the completion of each disk format/verify operation. Options on this menu are

Format Another
Main Menu
Check Disk
Quit To DOS

These options are self-explanatory.

## Rapid Format

RAPID FORMAT provides a way to very quickly reformat disks that are already formatted. If you have a batch of formatted disks that you desire to re-initialize, use this option. Instead of formatting the disk track-by-track, as is necessary in the case of a fresh, unformatted diskette, FormatMaster will skip the track formatting and re-initialize the FAT and directory sectors. The disk will then be just as if you did a full track-by-track format. If you use this feature in conjunction with CYCLIC = ONE (OR TWO), things tend to go rather rapidly, so there is a pause for a key press to allow time for swapping disks. THE VERIFY option may be used in conjunction with this feature.

A previously formatted disk may have bad sectors that were marked by DOS or FormatMaster when it was formatted. FormatMaster will detect this situation and automatically verify and update the disk FAT following the rapid format.

Note that it is NOT possible to rapid format an unformatted disk.

## Only Verify

ONLY VERIFY = ON turns on diskette verification but will not format the target disk. This feature is to be used ONLY with formatted disks and is useful in verifying the integrity of a data diskette. The diskette FAT will NOT be updated following the verification check. If FormatMaster discovers bad sectors on the disk, it is strongly advised to attempt to copy all the data from the diskette to a freshly formatted and verified diskette and then reformat and reverify the diskette.

## Define Drive Table

FormatMaster will attempt to detect how the floppy disk drives in your system are configured at runtime. Specifically, it is important to know the size (3½" or 5¼") of each drive and whether or not that drive is high capacity. Because there are so many configuration variants among PCs and ATs (and clones), it may not necessarily be true that floppy drives are sequentially designated from A to D. In fact, there are some systems with add-on 3½" drives (and attendant drivers) that are addressed as a letter that is higher than the hard drive letters in the system. For this reason, you may need to use the FM drive table setup feature to assure proper performance.

At runtime FormatMaster will automatically set the drive table according to information derived from the ROM BIOS or DOS (version 3.2 or later). In most cases this information will accurately reflect the status of the drives in your system, and it will be unnecessary to alter the drive table. There may be some cases, however, where the table is not set properly by FormatMaster, particularly in the case of add-on 3½" drives that use a DOS logical drive designator (i.e., B:,C:,D:) that is not in the usual sequence. The first time you run FormatMaster you should check the drive table settings to determine if they match the way your system is configured. If so, you need do nothing more (although it is okay to save the drive table, if you wish). Otherwise, use the menu to set the table properly and be sure to save the table before attempting to format any disks.

Floppy disk controllers can address at most up to four drives and access the available drives through numeric codes that range from 0 to 3. For example, just about every PC-type machine in the world will have a DOS drive A that is accessed by the BIOS as drive 0. Normally, DOS drive B will be accessed as drive 1, and so on. It should be clear by now that there are two ways of addressing system drives. You communicate with DOS about system drives using *logical* drive designators, which are alphabetic letters ranging from A to Z. DOS in turn calls the machine ROM BIOS to perform low level disk operations (read, write, format, etc.) and uses *physical* drive designators that are numbers. FormatMaster uses a user-defined drive table that keeps track of four important items — the DOS logical drive designator, the BIOS physical drive number, the size of the drive, and whether the drive is a normal or high capacity drive. A high capacity drive is one that supports both normal and high density diskettes. For example, the AT comes with a high capacity 5¼" drive as DOS drive A. It will format either normal (360K) or high-density (1.2MB) floppies. Many systems now have high capacity 3½" drives which format either 720K or 1.4MB diskettes.

Configuring the FM drive table is relatively simple. You select one of the four drives from the menu and then use the drive table parameter entry dialog window. This window will display the drive letter, drive size, and drive capacity edit fields to you. Use the mouse, Tab key, or Enter key to move through the fields. You can set the drive letter by pressing any letter key. The drive size and density fields are toggled by pressing the spacebar, + key, or – key. To accept and store your changes press the F10 key; otherwise, press the Esc key.

Once you have made the settings appropriate for your system, you can save them permanently to disk via the @ CONFIGURE FM main menu function. If in doubt about which logical drive matches which physical drive, you can experiment. FM contains numerous checks to determine if a hard drive is selected for formatting and will abort if this is the case. In addition, FM makes no calls to any DOS or BIOS function that writes to or formats a hard disk. One way to ensure that the drive table is properly configured and only floppy drives will be accessed is to perform an FM Check Disk operation with a formatted disk in each of the available system drives.

## Enter DOS

ENTER DOS provides the means to *shell out* to the DOS level to perform whatever operations desired. Entering the command EXIT from the DOS level will return you to the FormatMaster main menu.

## Check Disk

Selection of this item will bring up the drive selection submenu. Upon selection of a drive, FormatMaster will display a window that contains nearly the same information as the DOS CHKDSK utility provides. The Check Disk function is also available from the choice menu that is displayed during format/verify operations.

## Transfer System

System diskettes differ from data diskettes in that they contain a slightly different boot sector and they must contain the two DOS system files as well as a compatible copy of COMMAND.COM (or 4DOS.COM/NDOS.COM). FormatMaster will transfer system files to a freshly formatted disk, but you must ensure that it can access the DOS utility SYS.COM. You do this by ensuring that the file SYS.COM is in a directory whose name appears in your DOS path statement. For example, if your DOS files are in a directory named C:\DOS, then the directory pathname C:\DOS *must* appear in your path command. Please read the file PATHINFO.TXT for a more detailed explanation of the DOS PATH command feature. FormatMaster invokes SYS.COM to transfer system files to the newly formatted disk.

FormatMaster will attempt to locate SYS.COM, the DOS system files, and COMMAND.COM at runtime. If these files cannot be located, the Transfer System option on the main menu will be disabled.

4DOS and NDOS users should note that you have a choice of which command interpreter you wish copied to a floppy diskette when creating a system disk. If FM finds a copy of DOS COMMAND.COM anywhere in your DOS path, it will copy that file to the system disk. Otherwise, it will copy the file specified by the DOS environment variable COMSPEC=.

## Add Vol Label

ADD VOLUME LABEL provides a means for the entry of a diskette volume label prior to formatting any disks. Selecting this item results in the presentation of a window which will prompt for the desired volume label. Any valid DOS label name may be entered. If the entered label name contains a number as the LAST portion of the name, that number can be used to add sequential serial numbers to each formatted diskette. For example, if the entered name is SER-0001, FormatMaster will use SER-0001 for the first disk, SER-0002 for the second disk, SER-0022 for the 22nd disk, and so on. Any sequence of numeric digits ('0' to '9') may be used, but they must comprise the LAST portion of the label name. SER-0022X would not be used for sequential disk numbering. If an acceptable sequential number is entered, you will be asked if you wish to use the entered label for serializing disks. If the response is Y(es), then disks formatted using the Cycle Format option will be automatically sequentially serialized using the entered digit series as a base. The last serial number used can be saved via the "" menu item. When saving the current configuration to disk, FormatMaster will ask if you wish to save the current serial number. If the response is Y(es), then the current label name and sequence number is saved to disk. The next time FormatMaster is run it will use the next number in the sequence for cyclically formatting and adding the volume label. To turn off the addition of a volume label, simply press ESC or enter an empty string when prompted for the label. Entry of a label name that does not contain a proper sequence of digits at the end of the name will stop the sequential updating of the serial number.

If the toggle is ON, it may be set to OFF by selecting the menu item, deleting the current volume label, and returning to the main menu.

## Cyclic Format

CYCLIC FORMAT, when set to ONE or TWO, provides a means to rapidly format disks in TWO of your system floppy drives. Selecting this item will bring up a submenu from which ONE drive or TWO drive cyclic formatting may be selected. When entering the main format display, you will be prompted to enter two items for each drive. The first is the DOS drive letter designator. Make sure that this is a drive letter that you have set up in the SET DRIVE TABLE option. If your drive table setting indicates that the selected drive is a high capacity drive, you will be asked if you are going to be formatting high capacity diskettes in the drive. In this manner, any two drives in your system may be used for the cyclic formatting. After entering the pertinent information, you will be instructed to ready the selected drives for formatting. As soon as you press any key, the formatting of the two drives will start. You may terminate the formatting cycle by pressing the ESC key at any time. The formatting operation will continue to completion on whichever drive is active at the time the key is pressed, and a message will appear that informs you that the formatting will be terminated following the formatting of the current drive. If the Esc key is pressed a second time, the formatting will terminate immediately.

## Beeps

FormatMaster signals a number of operation completions with an audible beep. The beeps may be suppressed by toggling the Beeps toggle to OFF.

## Verify

VERIFY = ON switches the verification feature on and off. If set to ON, disk verification will be performed using the BIOS CRC verification routine. Every sector on the disk is checked, and all bad sectors are marked. FormatMaster will report bad sectors as cluster groups in the Advisory Message box. At the end of the verification, if there were any bad sectors found, FormatMaster will update the File Allocation Table (FAT) to lock out those sectors. Note that turning verification to ON will slow down the total formatting process. In some cases, FormatMaster may be more stringent than DOS and mark more sectors unusable. That will result in less total disk space available on a damaged disk, but is preferable to the opposite approach.

## Quit to DOS

QUIT TO DOS terminates FormatMaster and returns you to the DOS level. Alt-X will also terminate the program. Pressing the Esc key can also be used to exit FM but a confirmation window is presented first.

## Print Registration

PRINT REGISTRATION FORM brings up a window that will prompt you for the entry of registration information and then will create a registration form on your printer that can be mailed to New-Ware.

## @ Configure FM

To semi-permanently configure FormatMaster, use the @ CONFIGURE FM.EXE option. All the current toggle settings will be saved to disk so that each time you execute the program, it will start up with the toggles as set when you invoked this option. In addition, if you have started FormatMaster with an M (or m) on the command line, the monochrome (composite) monitor color settings will also be saved so that it will not be necessary to use the M on the command line.

Configuration settings are saved to a file named FMTMAS.CFG. The save configuration screen will indicate where FM intends to save this file and gives you a chance to save it to a specific location of your choice. When FM loads, it will look first in the current directory for FMTMAS.CFG. If it finds it, it will load it and proceed. If not, it will then search all the directories found in the DOS path for the file. It will load the first copy of FMTMAS.CFG it finds.

## The Escape Key

The ESC key can be used to exit from just about any menu or operation. It will also abort formatting with an advisory message. If the cyclic toggle is in effect, FormatMaster will terminate the operation after completing the format operation on the current disk. If the Esc key is pressed a second time, formatting will be aborted immediately.

## Error Handling

All detected errors are reported in the Error Message box, and error messages are accompanied by an audible beep. Bad disk clusters detected during verification are reported in the Advisory Message box.

## Terminate-and-Stay-Resident (TSR) Version

A special version of FormatMaster that installs as a memory-resident TSR is available directly from New-Ware. This version permits users to pop up FormatMaster from within a non-graphics application program and format floppy disks. POPFM requires only 6K of normal RAM and utilizes EMS, XMS, or hard disk to swap code/data in and out of memory. The activation key combination (hot key) is user-configurable. Users who order POPFM will receive the full FormatMaster package plus the special TSR utility for $30.

## Technical Notes

FormatMaster is written in a combination of Turbo Pascal 6.0 and assembly language and makes generous use of TurboPower Software's excellent Turbo Professional 5.23 package. Formatting is accomplished via INT 13h ROM BIOS service calls based upon parameters in the drive table or via the DOS block device control (IOCTL) function.

Disk verification during format does not perform a rigorous sector read/write type of verification and can be skipped if one is using decent quality diskettes. Note that the DOS device control function also utilizes the BIOS disk sector verification function. New-Ware has formatted and distributed thousands of low cost bulk-purchased diskettes that were formatted by FM without verification. The disk return rate to date has been practically nil.

The program was developed on an AST 20 Mhz Premium/386 equipped with one dual capacity 5¼" disk drive, one dual capacity 3½" disk, and a VGA monitor running MS-DOS 5.0 and QEMM 6.1. It has been thoroughly tested on the 386, an AST Premium/286, AST Bravo 286, and a 386 SX clone. MS-DOS 3.2, 3.3, and 4.01 were used on standard 360K/720K and dual capacity 1.2M/1.4M disk drives.

## Monitor Problems

Many laptop portables and those machines with a *composite* monitor can trick programs like FormatMaster into thinking that there is a fully functional Color Graphics Adaptor (CGA) on board. If you load the program and the cursor bar is not visible or the display looks strange, quit the program using Alt-X and reload it with an M (either case) on the command line.

**Example:**

```
FM m
```

This will force FormatMaster to use a set of monochrome attributes that should work fine on LapTop and composite monitors. Using the main menu Config command will save the current color configuration so that using the m on the command line will no longer be necessary the next time FormatMaster is loaded.

## Mouse Support

FormatMaster supports all the popular Mouse systems. The left button is used to enter a carriage return and the right button serves as an ESC key.

## BBS Access

The New-Ware Shareware Products BBS operates 24 hours per day, 7 days per week. Node #1 is dedicated to registered users only. Node #2 is open to all callers. Users registering via the BBS receive access to latest registered version within one day after leaving their credit card data. Users who register by mail or phone may also access the registered version by BBS, but you MUST log on and leave a comment to the sysop to the effect that you are a registered user and desire access. The author can also be contacted on CIS via PPN 71535,665 (please use EMAIL).

## FormatMaster Registration Form

Name: ______________________________
Company: ___________________________
Address: ____________________________
City: _______________________________
State: ____________ Country: _______________
Zip: ______________ Phone: ________________

**Remit to:**
New-Ware
8050 Camino Kiosco
San Diego, CA 92122-1820

Registering FormatMaster may be accomplished in one of the following ways:

1. Mail a check or money order in the amount of $20 payable to New-Ware. California residents please add 7.75% sales tax ($21.55 total). (Please send $30 for the TSR version).
2. Mail your Visa or Master Card number, expiration date, name, address, and phone number. No other credit cards can be accepted.
3. Call (619) 455-6225 and leave your Visa or MasterCard number, expiration date, name, address, and your phone number. If you call the voice number to register, you may get an answering machine. Have your card or the form printed by FormatMaster ready so you can leave your registration information quickly and accurately.
4. Call the Shareware Products BBS and register on line. The number is (619) 455-5226, N/8/1, 1200/2400 baud. Your credit card can usually be validated within one day, and this is the quickest way to get a copy of the registered version.

FormatMaster will be shipped first class mail within two days of receipt of a registration.

# HotDIR Plus

**Version 7.2**
**Copyright © 1992 by Robert Woeger**

## Introduction

HotDIR was the first and in my opinion is the best color-sorted directory utility. I originally released version 1.0 back in early 1987, followed by the "famous" HotDIR 2.1. Due to the change with hard disks larger than 33MB in DOS 4 and DOS 5, I rewrote HotDIR from scratch and released the HotDIR Plus 7.0 series. HotDIR Plus 7.0 and 7.1 were only released to a few people, and 7.2 is the first MAJOR release of HotDIR Plus.

## Scope of HotDIR

HotDIR Plus is a color-sorted directory utility. HotDIR Plus is unique in that it was the first to display files by color (based on what extension they were). It has options to sort your files in many different ways, and list them left to right or up and down, plus in 1, 2, 4, or 6 column format. I believe that once you've tried HotDIR you'll be hooked. Your files will *pop out* at you, and that is how the "Hot" in HotDIR came about. I originally wrote HotDIR for a friend of mine, Dr. John German, to help him see the different files by type (extensions) easier. He said, "that's Hot"! and I soon released the original HotDIR 1.0 to the world. I got such an enthusiastic response from HotDIR that I wrote HotDIR Plus.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the HotDIR Plus files in the directory \SECRETS\HDIRPLUS. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the HotDIR Plus files to a directory that is included in the search path.

## Using the Program

Type **HDIRPLUS /h** to get the help screen.

All commands are described on the help screen and appear as a forward slash followed by a letter of the option. You may use multiple commands when you invoke HDIRPLUS:

```
Format: HDIRPLUS [path] [filespec]
   [/option] [/option] ...
```

**Examples:**

```
HDIRPLUS \os2\system /c /2
```

This will display a 2-column directory format and clear the screen first. It will display all files in directory \os2\system.

```
HDIRPLUS
```

This will display all files in the default drive/directory and use the predefined options. Default or those modified through HDIRCNFG.EXE. HDIRPLUS /r Information about registering your copy of HotDIR Plus.

```
HDIRPLUS /h
```

List of all commands available for HotDIR Plus.

```
HDIRPLUS a: /p
```

Display directory of A: drive and toggle Pause at end of screen mode.

```
HDIRPLUS /p >prn
```

Sends HotDIR Plus's output to the printer without page pauses. You can send output to a file by placing >filename instead of >prn.

## Summary

Here is a summary of command switches used by HotDIR Plus:

| | |
|---|---|
| /H | displays on-line help screen |
| /R | displays information on how to register your copy |
| /N | No sort (unsorted raw file directory) |
| /F | Filename sort (factory default) |
| /E | Sort by extension |
| /D | Sort by date (oldest to newest files) |
| /S | sort by file size |
| /1 | 1-column display format |
| /2 | 2-column display format |
| /4 | 4-column display format (factory default) |
| /6 | 6-column display format |
| /C | toggles Clear screen at beginning of directory display |
| /P | Pause at end of screen toggle |
| /U | Up/Down file sort/display toggle — alternate is left-to-right display |

## The HDIRCNFG.EXE file

A program named HDIRCNFG.EXE is included to give you greater flexibility and control over the permanent HotDIR Plus settings. The above commands are great for overriding the default settings, but when you want to permanently change the colors used by HotDIR Plus, sort options, column displayed, etc., use HDIRCNFG to change HDIRPLUS.EXE. HDIRCNFG.EXE modifies the HDIRPLUS.EXE program itself and writes the changes inside the HotDIR Plus program. This was done to keep from having a separate configuration file for HotDIR Plus floating around. Due to a programming error on my part, HotDIR Plus 7.1 was clobbered whenever the HDIRCNFG program was run and the options saved. This has now been fixed, but I encourage you to make a backup copy of HDIRPLUS.EXE before running the HDIRCNFG utility.

Simply run HDIRCNFG and it will prompt you to change each setting. If you make a mistake, just do not save the changes, exit HDIRCNFG, and rerun.

If you want to remove an extension name and color, simply overwrite the extension name with another extension and color. If you want to just disable one extension, make sure you use blanks for the extension name. Make sure you move the other extensions forward and make the blank extension the last in the list. Otherwise, the blank extension will cause HotDIR Plus to stop searching for other colors and extensions that are after a blank entry.

HDIRCNFG is very versatile and powerful, but I recommend you get familiar with HDIRPLUS first before you change options. Most every option is able to be changed through HDIRCNFG.EXE. If you want to temporarily override an option, use a command switch when invoking HotDIR Plus. Also, you can change the number of lines to display per screen with HDIRCNFG. This is very useful for EGA & VGA monitors when in 43 or 50 line per screen mode.

HDIRCNFG uses standard BASIC COLOR command color numbers. It is recommended that you use colors between 0 and 15. 16–31 are blinking versions of the lower 0–15 colors. The most common colors numbers are

| | |
|---|---|
| 0 | Black |
| 1 | Blue |
| 2 | Green |

3 Cyan
4 Red
5 Magenta
6 Yellow
7 White
8 Gray
9 Light blue
10 Light green
11 Light cyan
12 Light red
13 Light magenta
14 Hi intensity yellow
15 Hi intensity white

## File Number and Size Limits for HotDIR Plus

There are some self-imposed limits with HotDIR Plus which are

- No more than 1,000 files per subdirectory. This is a reasonable number of files, and it helps keep HotDIR's sort and data areas to a reasonable size.
- File sizes up to 9,999,999 may be displayed without breaking HotDIR Plus's formatting on the screen. I chose this number as most files are less than 10 megabytes in size each. Note that the disk size may be over 1.6 gigabytes and HotDIR Plus should report the total disk free correctly. Only each file is limited to a maximum of 10 megabytes. If a file is greater than 10 megs, the file size should be displayed properly, but you will notice that since it takes extra digits to display, your 2-, 4-, or 6-column HotDIR Plus display may "wrap around" on the line where that huge file is. One-column displays will work best if you have a lot of huge 10 meg+ files in a directory.

## HotDIR Plus 7.2 Registration Form

Name: ____________________
Company: ____________________
Address: ____________________
City: ____________________
State: __________ Country: ____________________
Zip: __________ Phone: ____________________

**Remit to:**
Robert Woeger
2423 Hagerman St.
Colorado Springs, CO 80904-3228 U.S.A.
(719) 471-8306

Payment by:
☐ Check ☐ Money Order ☐ Purchase Order

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Upon registering, you will receive a registered version of HotDIR Plus, discounts/notice of software upgrades in the future, and full legal rights to use HotDIR Plus on your machine(s). Please read the terms of the software license and restrictions in the file HOTDIR.DOC. For information on quantity discounts, site licenses, and distribution of HotDIR Plus with your software, contact the author.

Regular HotDIR registration is $20 and includes a disk with the latest registered version of HotDIR Plus for DOS, software license, technical support, and news on updates. You may also register for $20 via CompuServe's Shareware Registration service (GO SWREG) and use Registration ID #364 to register your copy of HotDIR Plus. You will receive all the above including a disk with the registered version. Your CompuServe account will be billed the $20 and they will forward your name to us. For $15 you get all the above less the disk with the registered version of HotDIR Plus.

| | | |
|---|---|---|
| HotDIR Plus for DOS w/ disk each | $20.00 | $ ______ |
| HotDIR Plus for DOS no disk each | $15.00 | $ ______ |
| HotDIR Plus for DOS & OS/2 disk each | $25.00 | $ ______ |
| Colorado residents add sales tax | | $ ______ |
| TOTAL | | $ ______ |

Please send U.S. funds only.

Thank you for your business. Orders normally sent U.S. Mail 2nd day priority mail. All shipping & handling fees included in purchase price.

# Led's Change Directory

**A PC-DOS / MS-DOS Intelligent Directory Changer**
**Version 4.0**
**Copyright © 1991 by Keith Ledbetter.**

*Note from the Author:* McAfee Associates (the makers of SCAN and other anti-virus tools) are now distributing LCD. At press time, a new version of LCD, called MCD, is in the final stages of development. Contact McAfee Associates at (408) 988-3832 for further information about the LCD and MCD utilities.

## Introduction

LCD is a program designed to make changing directories much easier, especially when you have a large hard drive with lots of different partitions.

LCD is a work-alike to Peter Norton's NCD command, but with one significant difference. LCD works *across all drives* by maintaining the directory database of all drives on drive C:, instead of one database on each individual drive like NCD does. This means that you don't have to specify a drive letter when changing to a directory on a different drive.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the LCD files in the directory \SECRETS\LCDFILES. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the LCD files to a directory that is included in the search path.

## Changing Directories

LCD is used to quickly switch between directories across any of your hard drives.

**Syntax:**

| | |
|---|---|
| `LCD partial_ directory_name` | change to a directory |
| `LCD md directory_ name` | create directory; update database now |
| `LCD rd directory_ name` | delete directory; update database now |
| `LCD /scan [drivelist]` | scan drives and build directory database |
| `LCD /qscan [drivelist]` | scan drives (ignore dirs with extensions) |
| `LCD *` | bring up handy speed search and browse window |

You simply invoke LCD and give it the full or partial name of the directory you wish to change to. For example, if you have a directory named C:\PROCOMM\DOWNLOADS, you could switch to this directory with the command:

```
C:> LCD down
```

LCD will first try to do an immediate change to the directory name that you specify. If that change works, LCD simply exits immediately. If it fails, then LCD looks into your directory database and tries to figure out the directory name that you wanted by doing fuzzy comparisons. If more than one directory existed with the same partial name that you gave to LCD, it will display a scrollable dialog box on the screen containing all of the matches found. Simply use the arrow keys to highlight the path that you want to change to, then press <Enter>.

## Scanning Directories

The first time you execute LCD, you must specify the /SCAN parameter

```
LCD /scan
```

which instructs LCD to scan all of your hard drives and build a file called LCD.IDX in the root directory of drive C. You will need to periodically rescan all of your drives to add to the table any new directories that you have created.

If you only want LCD to know about certain specific drives (i.e., only non-network drives), you can specify a drive list after the /SCAN keyword. To have LCD only scan drives C, E, F, and G, you'd do this:

```
C:> LCD /SCAN CEFG
```

If you are sure that you don't have subdirectories which have an extension in their name (for example, C:\PROGRAMS.OLD\), you can specify /QSCAN instead of /SCAN. This causes LCD to scan almost 50 percent quicker than normal.

## Direct Database Updating

You can also use LCD to create and remove directories with immediate database updating. This means that you don't have to continually scan your drives to update the directory database. LCD will create or delete the directory specified and then update the LCD.IDX database at the same time. For example, to create a directory on drive F: and put that directory in the database, you would do:

```
C:> LCD md f:\test
```

This ability comes in especially handy for users of command shells that allow aliasing, such as CED and 4DOS. With these utilities, you can change the standard MD and RD commands to invoke LCD instead. With 4DOS, you would set up the aliases like this:

```
C:> alias md 'lcd md %1'
C:> alias rd 'lcd rd %1'
C:> alias cd 'lcd %1'
```

## The LCD Environment Variable

If you would rather have LCD keep its LCD.IDX database somewhere other than the root directory of drive C:, you can specify the filename you want used in the LCD environment variable. For example, if you'd like to keep the LCD.IDX file on drive D: in your UTILS directory, you would put

```
SET LCD=D:\UTILS\LCD.IDX
```

in your AUTOEXEC.BAT file.

***Note:*** Be sure to read the HISTORY.DOC file for changes that were made in this version.

## LCD Version 4.0x Registration Form

Name: ____________________
Company: ____________________
Address: ____________________
City: ____________________
State: __________ Country: ____________________
Zip: __________ Phone: ____________________

**Remit to:**
Keith Ledbetter
4240 Ketcham Drive
Chesterfield, VA 23832
(804) 674-0780 (6:00 p.m. - 10:00 p.m. Eastern Time)

Disk with branding utility
each ........................................ $15.00 $ ______
TOTAL ........................................ $ ______

Payment by: ☐ Check ☐ Money Order
Please make payment (in US funds) payable to "Keith Ledbetter."

Specify Disk Size: ☐ 5.25" 1.2MB ☐ 3.5" 1.44MB ☐ 5.25" 360K ☐ 3.5" 720K

Site, corporate, network, and reseller licenses are available for LCD.

# LIST

**A File Viewing and Browsing Utility**
**Version 7.7**
**Copyright © by Vernon D. Buerg 1983–92.**

## Introduction

LIST is a user-supported program. It is not public domain. You may use LIST and give it to your friends, but you may not sell it or use it in business without obtaining a license. See the last page for information about licensing.

You use LIST to display files on your monitor, line by line with the aid of scrolling, positioning, and filtering commands.

LIST PLUS has many new commands which go beyond usual file viewing and browsing. We will explain how LIST is used, and then how the new file management commands are used.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the LIST files in the directory \SECRETS\LIST. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the LIST files to a directory that is included in the search path.

| | |
|---|---|
| LICENSE | Licensing information and sample licensing agreement. |
| LIST DOC | Documentation file for LIST (may be PRINTed). |
| LIST HST | Version history, summary of changes to LIST. |
| LIST COM | LIST command, Plus version, all commands and features. |
| LISTR COM | LIST regular version. |
| LISTS COM | Small version of COM file without help screen. |
| DRLIST COM | Special CTTY/ANSI version of LIST. |
| DRLIST DOC | Documentation for DRLIST to be PRINTed. |
| COLOR PAT | Example DEBUG input to change colors. |
| -READ.ME | Registration information. |
| WHATSNEW | New command-line switches. |
| MAILER | Registration form. |
| LIST.ICO | Windows icon for LIST. |
| DIALER PAT | Example DEBUG input to change dialer COM port. |
| LISTMOD DOC | Documentation for customizing LIST. |
| PROGRAMS | List of other programs. |

## Varieties of LIST.COM

Before going into all of the ways in which LIST can be used to display files, let's look at the three different varieties of LIST so that you understand the capabilities of each. Then, we'll go on and define certain terms like redirection, piping, and filtering. In this way, you will better understand how you can use these things with LIST.

**LISTS.COM — small version**

- Runs in about 30K.
- Limited to smaller files (around 600K).
- Excludes the Alt-X (screen-saving) function.
- Excludes the Alt-G (goto DOS) function.
- The Help screen is minimal.

**LISTR.COM — regular version**

- Runs in about 80K.
- Handles files up to 16MB.
- Excludes the Alt-V (file selection) function.
- Excludes the Alt-I (insert filename) function.
- The Alt-W (windowing function) is minimal. That is, you get two equal-sized windows; there is no ability to change window dimensions.

**LIST.COM — PLUS version**

- Plus Alt-V file selection menu
- Plus file management functions like copy and delete

- Plus Alt-I hypertext-like file selection
- Plus a help screen for the file selection Alt-V functions
- Plus a second help screen for regular functions
- Plus the Ctrl-T telephone dialer

There is also a commercial version of LIST called LIST Enhanced. The main differences between LIST Plus and LIST Enhanced are

**The File Selection menu has**

- File tagging, including tag all, untag, retag, etc.
- Archive file extract and add/update commands
- A sweep command to execute a specified program with all of the tagged files
- A hardcopy (print) command for printing files with a user program
- Commands to change screen colors for all displayable items
- Shell to DOS
- Change video modes
- A directory tree display command
- The rename command that can rename directories
- A command to create new directories

**The viewer part has**

- The Alt-E command presents a menu offering up to 6 video modes including 132x25, 132x43, 80x43, etc., if the video adapter supports that text mode.
- Viewing of EBCDIC and ASCII files.
- Handling of fixed-length record files like database files at two or more times the usual speed.
- Optional number (on the left side) of each record.
- Command-line parameters to position to a given record number or to the end of file.
- Handling of files up to 32MB in size, larger upon request.
- The wrap option splits the line at a word boundary.

**In general the commercial version**

- Allocates memory more efficiently; can run in as little as 100K with all functions.
- Supports UltraVision video utility.
- Has options to use regular DOS input routines that take advantage of PCED, DOSKEY, or other keyboard utilities; or, to use an internal input routine which allows command-line editing.
- Has a customization program that can be used to set any of the LIST options or toggle and save them to a file or read the options saved in a file; the customization program can define other printer names, can define all the names of archive programs used.
- Has a 170-page spiral bound manual.
- Has a 6-panel Quick Reference card of all commands.
- Has one year of free updates.
- Has telephone, FAX, and BBS support.

LIST Enhanced may be purchased at computer stores or ordered directly from Buerg Software. Registered users of LIST receive a $20 discount.

The documentation file is marked with | before new and changed lines. It is marked with a double asterisk ** for LIST PLUS only features.

***Warning:*** LIST PLUS (LIST.COM) has the ability to DELETE files from your system!!

If you elect to use USE LIST PLUS, please read the section on the file selection menu found later in this manual.

***Note:*** Please see the file LIST.DOC for the sections omitted here on DOS terms and command lines.

## Starting LIST

To start LIST, type the command LIST at the DOS prompt followed by the name(s) of the files that you want to see. For example

| | |
|---|---|
| `C:>LIST CONFIG.SYS` | (displays file CONFIG.SYS) |
| `C:>LIST *.DOC` | (displays all DOC files in current directory) |

The *filename* is optional. If omitted, LIST PLUS will bring up a display of *all* files and subdirectories in the current directory. You may use the cursor keys to highlight the file that you would like to work on and press ENTER. Or, you may highlight any sub-directory entry and press ENTER to change to that subdirectory. You may also use LIST to display piped or redirected files.

To display a redirected file, use a < (less-than symbol) before the name of the file that was redirected and add the /S parameter to the LIST command.

For example, the output of the DIR command can be written to a file called XYZ, and then LIST can be instructed to read that file.

```
dir a: >xyz
list <xyz /s
```

To see a piped file, omit the filename, but supply the /S.

```
dir a: | list /s
```

Or, to LIST a file within an ARC archive:

```
ARC /p arcname.arc filename.ext |
   list /s
```

or

```
ARCE arcname filename.ext /p | list
   /S
```

Here, the ARC or ARCE program is invoked to extract the FILENAME.EXT file. The /P switch for these programs sends the output to the standard output device, and this output is piped to LIST. Once the file is displayed on your screen, you may use the cursor positioning keys to move around and see different parts of the file. There are also commands to search for text, print, split the screen, display other files, change colors, change the way the data is displayed, and many other operations.

## Exiting LIST

There are several ways to exit LIST depending on how you want the screen to look.

The F10 command returns you to DOS with the DOS prompt on the bottom line. The last page that was displayed by LIST is left on the screen.

The ESCape key also returns you to DOS without changing the screen. In LIST PLUS, the ESCape key is also used to cancel an operation or to exit the file selection menu.

The X command returns you to DOS and clears the screen. The DOS prompt is on the top line of the screen.

The Alt-X command uses the screen-saving feature. It returns you to DOS and displays the screen that you had before LIST was run.

## Entering Commands

You enter commands by pressing a single key or a combination of keys. There are often several ways to perform the same function with different keys. For example, D and PgDn both perform a scroll down one page function. This lets you pick the keys that you are most accustomed to.

When you press keys, they are entered into a keyboard buffer. Holding down a key can put many copies of that key into the buffer. This means that when you let up on a key, the program can still be processing input from the keyboard buffer. For example, by holding down the PgDn key, LIST scrolls down one page for each time the PgDn key is placed in the keyboard buffer. When you let up on the key, the buffer may not yet be empty and LIST will continue to page down. To make LIST stop when you let up on a key, you use the Alt-K key-ahead toggle.

Keyboard enhancement utilities, such as PCED, may also change the way that LIST reads the keyboard. With PCED, for example, pressing the ESCape key at a prompt is different. You do not see the / that DOS normally sends when you cancel an input line.

## Display Format

The monitor display is defined in terms of lines and columns. A typical monitor can display 25 lines of 80 columns each. LIST attempts to use the number of lines and columns for the monitor mode in use. For example, if the monitor is in 132 column mode, LIST displays 132 characters per line. If the monitor is set for other than 25 lines, such as 35, 43, or 50 lines, LIST displays that many lines per screen.

If you use the Alt-E command to change EGA/VGA modes, the EGA palette, cursor, and other settings are set to the DOS default values. LIST does not preserve fonts or palettes. The top line of the display is called the Status line. The bottom line is called the Command line. The remaining lines are called the primary display window and are usually lines two through 24.

## Status Line Format

The Status line has two formats. The default format is

```
|LIST lllll nnnnnnn +sss mm/dd/yy
   hh:mm - filename |
```

where,

| | |
|---|---|
| *LIST* | is the name of this program. |
| *lllll* | is the line number of the first line in the primary display window (under the status line). |
| *nnnnnnn* | is the line number of the last record of the file; if the last record of the file has not been read, this field shows the percentage of the file that has been read. |
| *+sss* | if displayed, this is the Scroll amount, in multiples of 10, corresponding to the number of columns that the display has been shifted to the right to view records longer than 80. |
| *mm/dd/yy* | is the file's creation date (not today's date); or |
| *dd-mm-yy* | is the file's date in European format. |
| *hh:mm* | is the file's creation time (not today's time). |
| *filename* | is the name of the file you are currently viewing. |

***Note:*** The date and time shown on the top line is not the current date. It is the date and time when the file was created.

***Note:*** See the file LIST.DOC for the visual example omitted here.

## Command-Line Format

The Command line has two formats. The default format is

```
|Command Keys: ^v-><- PgUp PgDn
   F10=exit F1=Help |
```

and the optional format after using the Alt-Z toggle is

```
|Command Toggles: h8kMpswTclJ
   F10=exit F1=Help |
```

where,

*command* indicates the current process:

| | |
|---|---|
| Command | function prompt; you are being asked to enter a command; enter the letter, or press the keys for the action to be performed. |
| Reading | the file data is being read. |
| Filter | the file data is being formatted for display. |
| Looking | the Scan/Find text is being searched for. |
| Scan ‘ | you are being asked to enter text to locate. |
| Find ‘ | in the file, up to 31 characters may be entered. |
| # lines? | you are being asked to enter a 1- to 5-digit number that is the amount of lines to skip. |
| Line #? | you are being asked to enter a 1- to 5-digit line number to which the display is to be positioned. |

*message* may be one of:

```
'*** Text not found ***'
the Scan/Text was not found in the
   file
'  *** Top of file ***'
the first line of the file is being
   displayed
'*** End-of-file ***'
the last line of the file is being
   displayed
```

**Toggles:** Indicates status of toggles: lowercase means OFF, uppercase means the option is ON.

| | |
|---|---|
| H | indicates that the hex Dump display option is in use. |
| b | the 'b' is replaced by a 7, 8, or * depending on which of those options is in effect. |
| K | indicates that the Keyboard flush option is in use. |
| M | indicates that tests for monitor retrace are not made. |

| | |
|---|---|
| P | indicates that Print is in use. |
| S | indicates file sharing option is in use. |
| W | indicates that the Wrap mode is in effect. |
| T | indicates that TAB characters are expanded. |
| C | toggles continuous scrolling. |
| L | indicates that the pre-loading option is on. |
| J | indicates that line feeds are added to lone carriage return control characters, and backspaces are handled. |

A sample Command Line might look like this, after looking for a word that was not found:

```
|Command *** Text not found ***
  Toggles: h*kMpswTclJ F10=exit
  F1=Help|
```

## Scrolling

You view different parts of the file by scrolling. That is, you use the cursor positioning keys up, down, left, and right to move the display one increment in that direction. For example, press the down-arrow key to move the display one line in the file (i.e., display the next line in the file).

The PgDn and PgUp cursor keys move the display one full screen in either direction.

Rather than press the up and down keys once for each line, you may use continuous scrolling. Use the C key to toggle continuous scrolling on or off. The default is off. When toggled on, pressing the up- or down-arrow keys results in a moving display. For example, when you press the down arrow, the next line of the file is displayed automatically every second or so. It is like holding down the arrow key. To stop the continuous scroll, press the spacebar.

The speed of the continuous scroll can be adjusted by pressing the + (plus) or – (minus) key while the display is moving. The plus key makes the display move faster, and the minus key makes it move slower. You can save the speed value by using the cloning command (see "Configuring LIST, Cloning," later on in this documentation). The speed is independent of the computer speed and can be adjusted from approximately 50 lines per minute to 1,000 lines per minute.

**Summary of scrolling commands:**

| | |
|---|---|
| right arrow | Move display right 10 columns. |
| left arrow | Move display left 10 columns. |
| down arrow | Display next line of file. |
| up arrow | Display previous line of file. |
| PgUp key | Display previous "page," 23 (or 41) lines back. |
| PgDn key | Display next "page," 23 (or 41) lines ahead. |
| C key | Turns continuous scrolling on or off spacebar — or any key, interrupts continuous scrolling. |
| + (plus) | Makes continuous scroll incrementally faster. |
| – (minus) | Makes continuous scroll slower. |

## Positioning to Lines

Each displayable line of the file is assigned a line number. The first line is assigned line number 1. The highest allowable line number is 16 million. In order to determine the last line number of a file, the entire file must be read. For this reason, the first time that the END (bottom) command is issued, it will take longer to process. This is not necessary if the file has been completely read (see the Alt-L preloading option). If the Wrap option is in effect, there is one line number required for each 80 bytes of the file's records. Thus, the line number does not represent the actual number of lines in the file. If the hex dump option is in effect, there is one line number required for each 16 bytes of the file. For example, an 80-byte line will be displayed as 5 lines.

You may position to a specific line number by using the Ctrl-Home or the # key. When Ctrl-Home is entered, you are prompted for the line number. Enter the line number. The display will now begin with that line number at the top of the screen. To position forward or backward, you may use the + (plus) or – (minus) keys. You are prompted to enter the number of lines to be skipped. The display resumes at the line number shown on the top (status) line plus or minus the number of lines that you specified.

When you change a filter option, such as Wrap, the line numbering changes. An attempt is made to retain the same file position, but the file may be repositioned at the top.

You may also reposition to the last *active* line by using the Alt-Y bookmark command. The last active line is one displayed after a Scan or Find command, or marked using the Alt-M or Alt-B commands, or the line set by the Ctrl-Y bookmark command. Up to ten bookmark lines may be saved by Ctrl-Y and recalled by Alt-Y.

**Summary of positioning commands:**

| | |
|---|---|
| Ctrl-HOME, or # | Prompts for exact line number to display |
| + (plus) | Prompts for the number of lines to skip for positioning further ahead in the file |
| – (minus) | Prompts for the number of lines to skip for positioning to an earlier line |
| Alt-Y | Reposition to the last bookmark |
| Ctrl-Y | Set new bookmark line number |

## Scanning for Text

There are three ways to initiate a search for text.

- First is the Find (\ or F) command. Use this command to search for text without regard to the case (upper or lower) of the letters.
- Second is the Scan (/ or S) command. Use this command to search an exact match. That is, the search is case sensitive.
- The third way to initiate a search for text is with the /F command-line switch. With it, you supply the text to search for, and LIST begins the search before displaying the file, and before asking for commands. The search continues through all selected files until a match is found, or the last file has been searched. This search is case insensitive.

Each search begins at the line displayed on the top of the screen and proceeds in a forward direction until the end of file. You may start a search that goes backward rather than forward by using the ‘ (left quote) key instead of Find (\), or by using ’ (right quote) instead of Scan (/).

To enter the Scan search text, type a slash (/) followed by up to 31 characters. The Scan text is displayed on the command line. The Scan is case sensitive. That is, lowercase Scan text will only match lowercase file text.

While the program is searching for the text, the bottom display line is changed to say `Looking`. Pressing ANY key while the search is in progress will terminate the search and display the message `Text not found` on the bottom line.

If the text is found, the line containing it is displayed as a high-intensity line (bright color) in the middle of the screen. The search text is displayed in reverse video colors on the highlighted line. The line where the found text is displayed depends on the setting of the “Find Row” (see “Configuration” section).

The display is scrolled left or right, as needed, so that the found text is visible.

If the text is NOT found, the command line is changed to say `'*** Text not found ***'`, and the rest of the display remains unchanged.

To find the next occurrence of the same text, use the A)gain command, or press the F3 key. If you wish to continue the search through all of the remaining files, use the Alt-A command. To find the previous occurrence, press the F9 key.

If the a keyboard enhancement program, such as PCED, is installed, the up/down cursor keys may be used to recall and edit previously entered Scan/Find text.

In a shared file environment, if the file changes while being listed, the file position may become invalid. Use the HOME command to ensure proper file synchronization, or use the Alt-S (share files) command again.

**Summary of scanning commands:**

| | |
|---|---|
| / or S | Scan for text with exact match |
| v or ’ | Scan for text going backward |
| \ or F | Find text with case-insensitive search |
| ^ or ‘ | Find case-insensitive text going backward |
| F3 or A | Find next occurrence of text |
| F9 | Find previous occurrence of text |

Alt-A Scan for next occurrence of the text, and continue on to the next file until the text is found. See the file LIST.DOC for sections omitted here on Marking and Extracting Lines, Printing, and Telephone dialer.

## Displaying Multiple Files

You can display more than one file at a time. LIST keeps track of up to 32 files at a time. You supply the filenames in the command line, through the Alt-F (get new filespec) command, through the Alt-I (insert filespec) command, and through the Alt-V file selection menu. The Alt-F command is disabled when viewing a piped file.

You can specify up to 16 filespecs on the command line. Each filespec can contain wildcards. Thus, several files can be selected via the command line. For example,

```
LIST *.DOC *.TXT
```

will display all files with an extension of DOC and TXT.

To display the next file, you use the Q or Ctrl-PgDn command. When the last file has been displayed, you can exit LIST by using the X, F10, ESCape, or Alt-X commands depending on how you want the screen to look when LIST ends.

To display the previous file, you use the Z or Ctrl-PgUp (control and PgUp keys together). The 1 command restarts the displaying of files with the first file.

The Alt-F asks you for a new filename, and you may enter a simple filename, or one that includes wildcards. These new filenames are added to the table of filenames that LIST keeps. Thus, you can use Alt-F for several different filenames, and use the Ctrl-PgDn and Ctrl-PgUp commands to move among them.

The Alt-I command allows you to select a filespec that is in the file you are viewing. You position the cursor to the filespec on the screen, and press enter to add it to the list of files to display.

The Alt-V command displays a list of files in the current directory. You move the cursor to a file that you want to display, and press enter to add that file to the list.

**Summary of file review commands:**

| | |
|---|---|
| Q | display next file, if any |
| Z | display previous file |
| 1 | display first selected file |
| Ctrl-PgUp | display previous file |
| Ctrl-PgDn | display next file |
| 1 | restart displaying with first file |
| Alt-F | prompt for new filename or filespec to display |
| Alt-I | Insert a filespec that is displayed |
| Alt-V | select a file from a menu |

## Windows

LIST can display data in one or two parts of the screen called windows. The Alt-W command toggles this split screen mode. Entering Alt-W while the screen is split restores the display to a single, full-sized display.

In the regular version, the screen is split in the middle. The top half of the screen remains the same, and the bottom half becomes the part of the screen where files are displayed. Thus, the top window becomes a scratch pad.

In the LIST PLUS version, you are allowed to split the screen at any point. After entering Alt-W, you are asked to position the cursor to the spot where the screen will be split. By placing the cursor in column 1, you can adjust how many lines will be used by the top and bottom windows. This would be a vertical, or one window above the other, split.

By positioning the cursor to the top line with the HOME key, you can then use the left- and right-arrow keys to adjust how wide each of the windows is to be. This would be a horizontal, or side-by-side, split.

You may display a different file in each window, and move between them. You use the Ctrl-V command to switch from one window to the other. You may also use the Review commands to display a different file in the second window. The Ctrl-V command has no effect unless you are viewing two or more files (i.e., a different file in each window).

Each window uses different colors. You can use the color commands to set the colors that you like, and then use the Alt-C command to save them.

***Note:*** Please see the file LIST.DOC for sections omitted here.

## Command Key Summary

**Cursor keys:**

| *Cursor key* | *Function* |
|---|---|
| left arrow | scroll left 10 columns |
| right arrow | scroll right 10 columns |
| up arrow | up one (previous) line |
| down arrow | down one (next) line |
| Enter | continue to next page |
| END | position to end of file (bottom) |
| ESCape | Exit program unconditionally |
| HOME | restart from first line (top) |
| PgUp | scroll up one page, 23 lines |
| PgDn | scroll down one page, 23 lines |

**Function keys:**

| *Function key* | *Function* |
|---|---|
| F1 | Displays the HELP screen. |
| F3 | Find NEXT occurrence of text after Scan or Find. |
| F9 | Find the PREVIOUS occurrence of text. |
| F10 | Exit to DOS. |
| For changing display colors: | |
| F2 | Change background color for Find/Scan text |
| F4 | Change foreground color for Find/Scan text |
| F5 | Change background color for main body of display |
| F6 | Change foreground color for main body of display |
| F7 | Change background color for top and bottom lines |
| F8 | Change foreground color for top and bottom lines |

## Letter Keys

The Letter key commands are mnemonic. That is, the letter in some way indicates what the command does.

| *Letter(s)* | *Function* |
|---|---|
| A | Find next occurrence of 'text' (Again) |
| B | Skip to end of file (Bottom) |
| C | Toggles continuous scrolling |
| D | Scroll down one page |
| F | Find 'text' regardless of case |
| G | Get new filename/filespec (also Alt-F) |
| H | Display the Help screen |
| K | Toggles keyboard key-ahead |
| L | Scroll Left 10 columns |
| M | Toggles Monitor retrace testing to eliminate snow |
| N | Down one (Next) line |
| P | Toggles the printing of displayed lines |
| Q | Quits current file and displays next file, if any |
| R | Scroll Right 10 columns command |
| S | Scan for exact text match, case dependent |
| T | Restart from first line (Top) |
| U | Scroll Up one page (23 lines) |
| W | Toggles the Wrap option for displaying long lines |
| X | Terminate, clear screen, and eXit to DOS |
| Z | Display the previous file |
| 7 | Toggles the 7-bit filter |
| 8 | Toggles the 8-bit filter |
| * | Toggles the star filter |
| + | Position a given number of lines forward |
| - | Position a given number of lines backward |
| ? | Displays the Help screen |
| space | Scrolls down one page |
| \ | text Find any case 'text' going forward |
| / | text Scan exact case 'text' going forward |
| ‘ or ^ | Find any case 'text' going backward |
| ’ or v | Find exact case 'text' going backward |

**Control- keys:**

The Ctrl- key commands are entered by pressing the Ctrl key at the same time as you press one of the following letter keys:

| *Ctrl- key* | *Function* |
|---|---|
| Ctrl-HOME | Position to a specific line by number |
| Ctrl-PgDn | Display next file |
| Ctrl-PgUp | Display previous file |
| Ctrl-left | Reset display to column 1 (i.e., scroll full left) |
| Ctrl-right | Scroll full right (to the end of the current line) |
| Ctrl-A | Scroll full left |
| Ctrl-C | Display next page, scroll down |
| Ctrl-D | Scroll right 10 columns |
| Ctrl-E | Display previous line, scroll up 1 line |
| Ctrl-F | Send a form feed control character to the printer |
| Ctrl-H ** | Hang up the telephone |
| Ctrl-N | Display previous file |
| Ctrl-P | Print the entire file |
| Ctrl-R | Display previous page, scroll up |
| Ctrl-S | Scroll left 10 columns |
| Ctrl-T ** | Dial a telephone number |
| Ctrl-U | Display previous file |
| Ctrl-V ** | Switch display windows |
| Ctrl-W | Display previous line, scroll up one line |
| Ctrl-Y ** | Save current line number as bookmark line for a–Y |
| Ctrl-X | Display next line, scroll down one line |

**Alt- keys:**

The Alt- key commands are entered by pressing the Alt key at the same time as you press one of the following letter keys:

| *Alt- key* | *Function* |
|---|---|
| Alt-A | Search for next occurrence of text, continue to next file until found. |
| Alt-B | Mark bottom line of display. |
| Alt-C | Copy options and setting to LIST.COM. |
| Alt-D | Write marked lines, or found line, to a file. |
| Alt-E | Toggle 25 or 43/50 line display with EGA or VGA. |
| Alt-F | Enter additional filenames to display. |
| Alt-G | Goto DOS temporarily to enter DOS commands. |
| Alt-I ** | Insert a filespec from the screen display. |
| Alt-H | Toggle Hex display mode. |
| Alt-J | Toggle the "junk" filter. |
| Alt-L | Toggle preloading of files. |
| Alt-M | Mark the line at the top of the display. |
| Alt-N | Toggle the Alt-X screen-saving feature. |
| Alt-O | Write marked lines, or found line, to the same file. Used by Alt-D. |
| Alt-R | Toggle the display of a ruler line on top line. |
| Alt-S | Toggle the file Sharing option. |
| Alt-T | Toggle the TAB control character filer. |
| Alt-U | Unmark lines marked by Alt-M and Alt-B. |
| Alt-V ** | Invoke the File Selection Menu. |
| Alt-W | Toggle split screen. |
| Alt-X | Exit to DOS and display the original screen. |
| Alt-Y | Reposition to the last *active* line (bookmark). |
| Alt-Z | Toggles the command line in the bottom line. The default is to display the cursor key usage. Using Alt-Z changes the bottom line to show the option switches "Toggles:" settings. |

## Limitations

- The LIST.COM program requires about 64K of memory. If more memory is available, it is used to store more of the file in memory. At least 80K is required to use the DOS Shell, and 9K more is required if the screen-saving option (on by default) is enabled.

- The line number is currently limited to 16 million.
- The file size is limited to 16 million bytes for ASCII files, and 4 million bytes for hex-dump files. Versions for larger files are available to licensed users.
- The review limit is 32 files for the regular and Plus versions. The bookmark limit is 10 entries.
- PC DOS Version 2.0 or later is required. DOS version 3.0 or later is required for file sharing.
- ANSI.SYS is *not* required.

## THE BUERG UTILITIES Registration Form

Name: ____________________
Company: ____________________
Address: ____________________
City: ____________________
State: __________ Country: __________
Zip: __________ Phone: __________

**Remit to:**
Buerg Software
139 White Oak Circle
Petaluma, CA 94952

(707) 778-1811 from 10a.m. to 7p.m. Pacific time, M-F.

CompuServe: 70007,1212 Forums: IBMAPP, IBMCOM, IBMHW, IBMNEW, IBMPRO, IBMSYS, and HAMNET

Data/BBS: (707) 778-8944, -or- (707) 778-8841, 24-hour bulletin board systems FAX: (707) 778-8728

The registered version of LIST Plus is $37 plus appropriate postage. You receive a disk with the latest version, notification of updates, and a printed manual.

The commercial (retail) program LIST Enhanced is $99 plus shipping and may NOT be copied or distributed.

Copies of the "Buerg Utilities" disk set for personal use may be obtained on 5.25"/1.2MB or 3.5"/1.44MB diskettes for $60 plus shipping.

The above products may be ordered by sending check, money, or credit card information with the "Mailer" included with this file.Or you may order with a credit card by phone.

Please send me the following:

| | | |
|---|---|---|
| LIST Plus, Version 7, registration (disk, manual, notification of major updates) | | |
| each | $37.00 | $ ______ |
| Shipping and Handling each: | | |
| U.S. & Canada | $3.00 | $ ______ |
| Other | $5.00 | $ ______ |
| LIST Enhanced, Version 1.8, commercial package (dual media, 160-page manual in slipcase, quick reference card) | | |
| each | $99.00 | $ ______ |
| Shipping and Handling each: | | |
| U.S. | $6.00 | $ ______ |
| Canada | $12.00 | $ ______ |
| Other | $24.00 | $ ______ |
| Complete Buerg Utilities (includes registered LIST Plus, 1.2MB/1.44MB disks only) | | |
| each | $60.00 | $ ______ |
| Shipping and Handling each: | | |
| U.S. & Canada | $3.00 | $ ______ |
| Other | $5.00 | $ ______ |
| California sales tax | 7.5% | $ ______ |
| TOTAL | | $ ______ |

Specify Disk Size: ☐ 5.25" 360K ☐ 3.5" 720K ☐ 5.25" 1.2MB ☐ 3.5" 1.44MB

Payment by:
☐ Check ☐ Money Order
☐ Visa ☐ Master Card ☐ Discover
Check or Money Order payable to Buerg Software in U.S. funds.

Authorized Signature: ____________________
Expiration Date: ____________________
Card Number: ____________________

# Patriquin's Hard Disk Utilities

Copyright © 1987–1992 by Norm Patriquin, Patri-Soft

## Overview

The Patri-Soft Utilities were developed out of my own need for enhancements to commands that are provided with DOS. I found that DOS commands did not provide me with functions I commonly needed or options I felt were obviously necessary for normal system maintenance. For example, the DOS COPY command fails to warn you before overlaying a file with the same name. I was hurt by this flaw once too many times and decided to develop my own alternative.

Basically, the Patri-Soft Utilities all function in a way you would expect them to. For instance, PCOPY works just like DOS COPY. Without any additional parameters, it will copy files like DOS, except some safeguards are implemented. To override the safeguards, or to use an additional feature, specify additional parameter switches.

***Note:*** The following utilities are included with this book: PCOPY, PDAILY, and PDELETE. Others are mentioned on the order form. Contact Patri-Soft for additional information on the utilities not included here.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the Patriquin files in the directory \SECRETS \PATRUTIL. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the Patriquin files to a directory that is included in the search path.

# PCOPY

Copyright © 1987–1992 by Patri-Soft

## Introduction

PCOPY is an advanced replacement for the DOS COPY command — one of the most used of the DOS commands. Unfortunately, the DOS COPY abilities are very limited and are not sufficient for many disk maintenance chores. PCOPY is similar to the DOS COPY command in that it copies file data between disks and hard disk directories. In addition, PCOPY provides intelligent file selection and processing options. PCOPY options allow you to use a single PCOPY command to perform a function that would require many complex DOS COPY commands.

PCOPY also provides a safer way to copy files than that provided by the DOS COPY command. It allows you to copy only newer files or update a directory with the contents of another. PCOPY will never allow you to overlay a file unknowingly.

PCOPY allows you to specify standard DOS wildcard pathnames in the source file specifications, and standard pathnames for the target directory specification.

PCOPY attempts to anticipate your needs and then help you accomplish your intent. For instance, if PCOPY determines that the target directory name does not exist, it will ask you if it is to be created. If you respond <YES>, PCOPY will create the directory. With the /SAve option, if you are about to overlay a file with another file with the same name, you will be asked if the older duplicate file is to be renamed with a version number.

The PCOPY.DOC file contains a section outlining all changes for all releases of PCOPY. If you already have a current release, this file will let you identify the changes since the version you have.

This archive should contain the following files:

| | |
|---|---|
| PCOPY.EXE | PCOPY execution file. |
| PMOVE.BAT | A batch file to be used with PCOPY. Causes files to be deleted from the source location after they have been copied. |
| PCOPY.DOC | PCOPY documentation file. Use the following command to print PCOPY documentation.<br>`COPY PCOPY.DOC PRN:` |
| PCREADME | A file to explain the contents of PCOPY system. |
| ORDER.FRM | Text form for ordering PCOPY. Use the following command to print this file.<br>`COPY ORDER.FRM PRN:` |

## The PMOVE Option of PCOPY

PMOVE is an option of the PCOPY command you can use to move files between disks and directories on disks. PMOVE combines the functions of DOS COPY and DELETE commands to simplify the moving of files. In addition to just moving files, PMOVE also allows you to specify selection criteria to better qualify the files to be moved.

PMOVE automatically determines the environment in which the move is requested and determines if the data must be physically moved or if it can be moved by simply renaming it. The file will be moved by renaming it if the move is to another directory on the same disk device. This is much faster than physically moving the file. In addition, it reduces free space fragmentation, and allows you to move very large files between directories when they could not otherwise be moved with DOS COPY because of insufficient space.

The only difference between PCOPY and PMOVE is that PMOVE removes the file from the source location after it has been successfully copied to the target location.

PMOVE is implemented as a .BAT file that invokes PCOPY with the /X parameter.

## PCOPY Features

- Copies files to and from any disk or directory.
- Allows you to save older versions of files with new names before replacing them.
- Moves files by copying them or renaming them to the new location.
- Deletes the original file if /X is specified.
- Creates target directories if needed.
- Makes sure the DOS archive flag is set correctly for each file as it is moved.
- Preserves the DOS date and time for each file moved.
- Allows you to pause processing at any time by pressing any keyboard key. Once stopped, the program can be terminated by pressing <ESC>.
- Provides file selection and processing control options to tailor the move process to your own needs.
- Allows commands to be tested so you may be sure that commands are specified as you desire.
- Copies updated files to a special disk or directory.
- If there is not sufficient space on the current target disk, PCOPY allows you to place another disk in the drive and continue processing. This lets you copy groups of files larger than disk size to other computers.
- Start with a specific file in a directory.
- End with a specific file in a directory.
- Process files based on the date stored in the directory entry.
- Warns you before overlaying a file unless specific parameters indicating otherwise are specified.
- Checks target disk for available space before starting to copy files. This saves time when the copy will not be able to be completed.
- Gives you constant status of the progress of the command.
- Runs with windows or using standard DOS screen output.

- Format new floppy disks as they are needed by pressing <F3> at any prompt.
- Process one directory, a single directory subtree, or all directories on a disk (/S).
- Allows you to indicate that multiple input disks are to be used and prompts for the next one after each disk is processed.
- Create a series of output disks filled efficiently with files from the source location. DOS ERRORLEVEL set to indicate error conditions. PCOPY may be used in .BAT files to copy files, and you may confirm results are good.
- Special wildcard matching allows you to select all files with specific text in filename.

## PCOPY Command and Parameters

PCOPY lets you specify command parameters to qualify the copy criteria. These parameters are optional. If no parameters are specified, PCOPY performs in a manner similar to the DOS COPY command, except that it prompts you before overlaying files. The command format and parameters are described as follows.

```
PCOPY from_file [to_file]
   [(/ parameters)]
```

*[from]* Enter the DOS file specification for the directory and filenames to be moved. The filename portion of the file specification may contain wildcard characters documented by DOS (* and ?).

You may also specify special pattern selection for text. For example, *DISK*.* would select all files that have the characters **DISK** anywhere in the first portion of the filename. *DI*S*K*.* type specification also works.

You may optionally supply a file containing a list of input filenames. Create a file containing a filename or complete pathname on each line. Specify the name of this file as the source filename to PCOPY and add the @ character before the name. The list of files will be processed as if they were all specified on the input command line.

For file list processing, PCOPY understands the output of the DOS DIR command. You may use the DOS redirection feature to direct DIR output to a file, edit the file to add or remove entries, and then provide this to PCOPY as input. For example:

```
DIR A: >DIRLIST
(edit dirlist)
PCOPY @DIRLIST B:
```

*[to]* Enter the DOS file specification for the directory and filename to be copied to. If the *to* specification is not entered, files will be moved to the current directory.

If a single level name is supplied as the last part of the target file specification, PCOPY will search for a directory that matches the specified name. If it does not exist, PCOPY will prompt you to determine if the name is a filename or a directory name. To avoid this prompt, append the name with a \ if it is a directory name, or a . to indicate it is a filename. For example:

```
PCOPY *.* C:\DIRNAME\ PCOPY
    *.* C:\FILENAME.
```

/4 4DOS support. PCOPY will copy and insert the appropriate 4DOS descriptions when copying files.

/A Archive. Selects only files that have been marked updated by DOS. This flag is updated by DOS whenever a file is processed with an intent to change or create new data.

/AF Assume that the target specified is a File. This will suppress PCOPY's prompt if the target does not exist.

/AD As above, except Assume Directory.

/AP APpend. All source files selected will be concatenated and placed together in a single output file. The output file will be given the name of the first or only selected input file, or the target filename specified.

If a target filename already exists when append is specified, all source files copied will be appended to the target file. To cause an already existing target file to be replaced by the copied files, also specify the /R parameter.

To combine 3 files into one with /AP, use a command like the following:

```
PCOPY FILE1,FILE2,FILE3
   NEWFILE /AP/R
```

/AZ Append text files removing Ctrl-Z characters. Like /AP except Ctrl-Z characters will be removed from the end of all but the last file copied.

/B Backup. When specified, the DOS ARCHIVE attribute is reset after a file has been copied.

/BA BAtch processing. Normally, PCOPY will pause if you press any key during processing. This can interfere with programs that store keystrokes in the keyboard buffer for subsequent activity. The /BA parameter sets "no-windows" mode and disables the "key press" check to leave stored keystrokes undisturbed.

/BW Black and white. Forces PCOPY to use monochrome colors even when it detects a color display environment.

/CD:xxx This parameter allows PCOPY to invoke any command of your choice for each file PCOPY selects. With this parameter, PCOPY will not copy your data. It simply becomes a file selection utility.

To use this parameter you must supply the command PCOPY is to execute. To be compatible with DOS command-line option specifications, you must use special substitution characters to build a command to your specifications.

The following special characters are used:

# Is replaced with the full pathname of the file to be processed.

$ Is replaced with the filename of the file selected.

@ Is replaced with the target filename PCOPY would have copied the file to, if the COPY operation was active.

[ Is replaced with a single space character.

] Is replaced with a backslash character (\).

} Is replaced with a slash (/) character. This must be used to allow a command to include / parameters, since PCOPY will interpret a slash as one of its own parameters.

PCOPY can be used to drive an EDIT command with variable parameters. The following is an example:

```
PCOPY *.c/d:)1jan89
/cd:EDIT[#[}F[}pf:]log
```

Assuming file C:\COMP\TEST.C was selected, the following command would result.

```
EDIT C:\COMP\TEST.C /F
/PF:\LOG
```

To execute the command, PCOPY invokes a second copy of the DOS command processor. When the /CD parameter is used, PCOPY forces operation without windows to eliminate screen flashing between commands.

/CF:path This parameter alters the meaning of the /CD parameter. When specified with a full DOS pathname (including the filename), it causes PCOPY to write all generated commands to the specified file. The commands will not be executed by PCOPY. This allows you to generate a .BAT file that can be modified and executed at a later time.

For example, to generate commands and place them in batch file DOIT.BAT:

```
PCOPY *.c/d:)1jan89
   /cd:EDIT[#[}F[}pf:]log
   /CF:DOIT.BAT
```

| | |
|---|---|
| /C2 /C3 | Specify /C2 or /C3 to use different colors when PCOPY is used with windows operation. To make this color change permanent, specify the parameter in the DOS environment variable. |
| /D:[(]xxx date | Select files based on their last update date. This parameter may include a date or a number of days value, and optionally a condition modifier that changes the use of the date or number of days specified.<br>When xxx contains a valid date, the date is compared to the last update date of the file, contained in its directory entry. The file is selected if the condition specified by the condition modifier is met. The condition modifier meanings are<br>( File date is older than that specified.<br>) File date is newer than that specified.<br>= File date is equal to that specified.<br>If the condition modifier is not specified, ) is assumed. Date values may be specified in any of the following formats:<br>MM/DD/YY 12/31/80<br>MM-DD-YY 12-31-80<br>DDMMMYY 31DEC80<br>If xxx contains a numeric value instead of a valid date, that positive number is used to calculate a date value. The new date value becomes the current date minus the number of days specified by the numeric value in xxx. Then that date is used as in the calculation above.<br>The /D: parameter may be specified twice to select files between two dates (date ranges).<br>/D:(1FEB90/D:)1MAR90<br>When two dates are specified, a file's date must qualify selection based on both date parameters. Specify the /DO parameter to indicate an OR condition between the dates. This will allow any file meeting either date parameter to be selected for processing. |
| /D- | Delete all files on target drive before starting copy. |
| /D2 | Add files to those already on the first output disk. Then delete all files from subsequent disks provided before copying to them. Use /D2 when you want to append files to a set of already existing disks. This parameter allows you to continue backing up to a partially used disk and then reuse disks containing old unwanted backup files. |
| /DC & DCA | Directory Copy. PCOPY will create a duplicate directory on the output disk for every directory processed on the source disk. Directories are only created when files are to be copied to them. See also the /S option. This is useful for copying all files and directories from one disk to another.<br>Specify /DCA (Directory Copy-All) to copy all directories from source to target drive, even when no files will be moved to the new directory. |
| /DL | Same as /DC except that only the last node of the source directory is copied to the target pathname specified. |

| | |
|---|---|
| /DO | Specify when two /D: date parameters are used. Indicates either date parameter may be met to qualify a file for processing. See also the /D: parameter. |
| /DR:xxx | Drive. Use to process multiple disk drives. /DR assumes the presence of the /S parameter, which indicates to process all directories on the source disks. If you are using PCOPY for backup, this lets you process all disk drives with a single command. |

The following command will copy all modified files from hard drives C:, D:, and E: to the floppy disk in drive A:. The /D2 parameter will ensure each new output disk is cleared of all files before it is written to.<shift right 44> <alt T><alt C><tab 3><down 6>t<enter>

```
PCOPY *.* A: /DR:CDE/A/D2
```

| | |
|---|---|
| /DS | Same as /DC except that only the current directory level and all its subordinate levels are copied to the target directory. The target directory may contain a pathname with multiple directory levels, so you can copy one directory structure below another directory structure. |
| /DX | When target names are single level names with no extensions, PCOPY will ask you if the name refers to a directory or to a new file to be created. If you specify this parameter, PCOPY assumes names without extensions are new directory names. |
| /DY | Synchronize source and target directory files. Before copying the first file to any directory, PCOPY will determine if any files exist in the target directory but not on the source directory. If so, PCOPY will remove those files from the target directory. Use this parameter when you want to update one directory or disk to be exactly like another. |

***IMPORTANT!!***

PCOPY has the ability to synchronize an entire drive by traversing the directories, deleting directory trees and files that are on the target but not the source. This feature is utilized by a careful combination of switches. Relevant switches are /S, /DC, and /U. Situations vary, so we have forced the /TF (Test First) on PCOPY when you use /DY. To override this Test, use the /TO parameter. BE SURE THE TEST PRODUCES THE DESIRED RESULTS BEFORE DOING IT FOR REAL! If you have any questions, or need a customized command line, feel free to call the Patri-Soft technical support line, (714) 352-2820.

To synchronize two drives:

```
PCOPY C:\ D:\ /DY/S/DC
```

To synchronize two directories:

```
PCOPY C:\UT D:\UT /DY
```

To synchronize a drive and a target dir tree:

```
PCOPY C:\ D:\BACKUP /S/DC/DY
```

| | |
|---|---|
| /E | Select and move only files that exist in both the source and target directories. This allows you to replace all duplicate named files. |
| /EQ | Designed for batch file operation. Use this parameter to prevent PCOPY from requiring you to press a key to terminate processing (Exit Quick).<br>***Note:*** Previous versions of PCOPY supported this option using the /EX. Parameter. |
| /EX | Exclude files. The exclude facility allows you to provide a list of pathname and filename specifications that are to be excluded from PCOPY processing. Since searching through data on your hard disk requires extensive processing and disk read time, excluding files from the process will save you time, and will reduce the number of false matches displayed for your requests.<br>The exclude facility is implemented with two actions. First, you will need to create an |

exclude specification file and place it in the same directory where PCOPY is installed on your hard disk. This file will contain the names of all the files and paths to be excluded. Name the file PCOPY.EXC.

You may specify DOS pattern names to exclude groups of files with a single specification. Like DOS, use the '*' to indicate the last part of a filename or filename extension may be any set of characters. Also like DOS, use the '?' to indicate that any single position of a filename may be any character.

In addition to DOS, two additional pattern features are available. You may place an '*' anywhere in the filename to indicate any set of characters may exist in this point in the name. For pathnames, you may place an '**' at the end of a directory path specification to indicate the specification applies to the specified directory and all subordinate directories. Using '**', you can exclude an entire path with a single specification.

Following are examples:

1. *.EXE
2. *.COM
3. TEST*.*
4. *TEST*.*
5. DO?FIX.*
6. PC*XX.*
7. C:\DOS\*.*
8. \TEST*.*
9. D:\GAMES**\*.*

Explanations for the above examples:

1. Excludes all filenames with the extension .EXE from processing.
2. Excludes all filenames ending with extension .COM.
3. Excludes all filenames beginning with the letters "TEST" from processing.
4. Excludes all filenames having the characters "TEST" anywhere in the first portion of the filename.
5. Exclude files having 'DO' as the first 2 characters of the filename, and 'FIX' as the 4th, 5th, and 6th filename characters. The 3rd character of the filename may be any character.
6. Exclude files having a name beginning with the 2 characters PC and where the first portion of the filename ends with the characters 'XX.' Any 0 to 4 characters may exist between the 'PC' and 'XX' in the first portion of the filename.
7. Exclude all files from the DOS directory on drive C:.
8. Exclude all files from the TEST directory existing on any drive searched.
9. Exclude all files from the D:\GAMES\ directory and also exclude all files and directories subordinate to the GAMES directory.

Activating exclude —

Exclude will only be functional when the /EX command parameter is specified. When /EX is specified, you must have a valid exclude file. PCOPY will not continue without one. If you wish exclude to be active during all executions, specify /EX as part of the PCOPY= DOS environment variable. See later in the documentation for a description of this feature.

| | |
|---|---|
| /F:file | First. Do not start processing until the specified filename is encountered in the source directory. This parameter is useful for restarting a move or copy that has been terminated. You may specify DOS pattern name characters * and ? to identify the file to begin processing. |
| /FI Fill. | Fill output disks as efficiently as possible. Normally, files are copied in the sequence in which they are found in the source directory. If a large file cannot be copied onto the output disk, a new disk is requested even though there may be enough space left on the output disk to hold other smaller files. The /FI parameter causes files to be copied in descending size sequence. When the next file will not fit on the output disk, PCOPY searches for another file that will fit on the disk before requesting a new output disk. When a new disk is placed in the drive, PCOPY begins processing the larger files again. Eventually, all files requested will be copied. |
| /FL | PCOPY's automatic disk formatting senses the drive type and will format the disk, if necessary, accordingly. If you want to format a low density disk in a high density drive, use this parameter. |
| /FO | Format Off. This turns off PCOPY's automatic disk formatting/checking. |
| /IFA: pathname | Check to determine that the source file does not already exist in an alternate directory before processing it. If it does not exist, processing will continue. Use just like IFP specified as follows. |
| /IFP: pathname | Determine if the source file to be copied already exists in the pathname specified by IFP. If it does exist, process the file. If it does not exist, skip the file. For example,<br>`PCOPY *.BAT \newdir /IFP:\testdir`<br>copies files from the current directory to \NEWDIR only if each file already exists in \TESTDIR. |
| /L:file | Last. Stop processing after the specified filename is encountered in the scan of the directory. The file specified will be processed if it matches selection criteria. You may specify DOS pattern name characters * and ? to identify the file to begin processing. |
| /LG | LoG activities. PCOPY will create a disk log of all actions it takes. This is useful if you are using PCOPY as a backup system. The log will help you locate the disk on which a file has been backed up.<br>The log file will be named PCOPY.LOG. For DOS versions 3.0 or later, the log will be created in the same directory from which PCOPY is executed. Other DOS versions will put the log in the current directory.<br>If the log does not exist, PCOPY will create it. If the log does exist, PCOPY will add to it. To begin a new log, delete PCOPY.LOG before running PCOPY. |
| /M | More. Stop the screen from scrolling past the end without user intervention. |
| /ME | MErge. Works like using /U and /X parameters to move the most current files from one directory into another. In addition, it deletes all the files that are not moved. This effectively merges the most current files from two directories into a single directory. |
| /MU | MUltiple. Process multiple input disks. After the first input disk is processed, PCOPY will prompt you for additional disks. Indicate <Y> to process more disks or <N> to terminate processing. |
| /N | New files. Only move files that do not already exist on the target directory. If the file already exists, do not move it. |

| | |
|---|---|
| /NA | No Archive. Only process files that do not have the DOS ARCHIVE attribute set in the directory entry. With this parameter specified, PCOPY will only process those files that have already been backed up. Files not backed up will not be processed. |
| /NP | No Pause. Use when you have made the /P parameter the default action with the SET PCOPY=/P command in the DOS environment, and want to override it. |
| /NW | No Windows. PCOPY will not use windows during processing. All screen output will be done through standard DOS facilities. |
| /O: filename-list | Omit. Specify a list of filename patterns that are to be omitted (excluded) from PCOPY processing. If multiple filename patterns are specified, separate them with commas. For example:<br>`/O:*.BAK,*.EXE,TEST*.*` |
| /P | Pause before each file is moved and ask for approval to process it. |
| /QA:x | Quit After process x number of files. This can be useful if you only want to process a certain number of files. |
| /R | Replace any duplicate files on the target disk or directory. PCOPY will replace files regardless of file dates and times. |
| /RD | Reset Date. Normally, PCOPY retains the original date and time of a file when copying it. With /RD specified, the copied file will have the current date. |
| /RE | REgister the program. Use this option to register your copy of the PCOPY utility. You will need to have the registration number and code that is sent to you after you request registration.<br>When you specify this parameter, part of the registration screen also allows you to modify the windows option permanently. If you do not want windows during processing, use this option to change the windows program option. You do not need to be registered to change this information. |
| /RO | Read Only. PCOPY is just like DOS COPY when it enounters RO files. PCOPY will copy the file, but does not copy the Read Only attribute. Using /RO places the Read Only attribute on the target file. |
| /S | System. If a source directory is supplied, that directory and all subordinate directories will be processed. If no source directory is specified, all directories on the disk will be processed. |
| /SA | SAve. Instead of overlaying already existing files with duplicate names, PCOPY renames the file in the target directory with a version number before the new file is copied. This ensures that no data is lost. This facility can be used to save successive versions of a file in a directory.<br>If the file has an extension name, the last two characters of the extension are replaced with a number from 01 to 99, depending on the next available number. PCOPY determines the next available number by looking in the directory for other files with similar names. /SF System Files. Normally, PCOPY skips files marked with the DOS SYSTEM attribute. Using /SF causes SYSTEM files to be selected for processing. |
| /SHx | File sharing options.<br>/SHA Ignore file sharing and copy all files.<br>/SHR Allow files opened for read to be copied.<br>/SHN Disallow all file sharing. |
| /SP | Split files. /SP indicates PCOPY is to split a file onto another disk if |

there is insufficient room on the first disk.

/SP:nnn Split a file nnn characters into the file. If disk space permits, both parts of the split file will remain in the target directory.

/SPL:nnn Split a file at line number nnn. This can be useful for breaking a file into parts for editing.

Split File Parameters:

You may split a file into two or more parts with PCOPY. This is useful for copying files larger than a diskette will hold, or for dividing files into smaller parts for editing. PCOPY identifies the file parts by replacing the second character of the file's extension with a number. For example, PCOPY.EXE could be split into PCOPY.E1E and PCOPY.E2E.

The time field of the first part of a split file holds the replaced extension character. For this reason, PCOPY can only automatically rejoin an unmodified split file. Just reference the first part of the split file, and the rest will be copied. You may also rejoin files using DOS COPY or PCOPY explicit commands on each part of a split file, as follows:

Rejoining Split File with DOS COPY:

```
COPY /B FILENM.E1E+
   FILENM+E2EFILENM.EXE
```

Rejoining Split File with PCOPY:

```
PCOPY A:\FILENM.E1E,A:
   \FILENM.E2E
   FILENM.EXE/SP
```

***IMPORTANT!*** PCOPY cannot split files into more than NINE parts.

/SS:nnn Terminate processing when the source drive's freespace reaches the value specified by the /SS parameter. Use when moving files from one drive to another using the /X parameter. The command PCOPY C:\*.\* D: /X/SS:1000000 will cause PCOPY to move files from drive C: to drive D: until the freespace on source drive C: is greater than or equal to 1,000,000 characters.

/ST:nnn Terminate processing when the target drive's freespace becomes less than or equal to the value specified by the /ST: parameter. The command PCOPY C:\*.\* d: /ST:500000 will cause PCOPY to copy files to target drive D: until drive D: has less than or equal to 500,000 characters freespace.

/SZ: [(] [)] nnn Select a file based on its size. The ( indicates that files less than the indicated size are to be selected. The ) indicates to select files greater than the size. If neither is specified, ( is assumed.

/T Test. Test the command without actually updating any files. Issue messages to indicate what will happen if the command is executed without the /T parameter.

/TO When using the /DY parameter, PCOPY forces /TF. Use this to override the test.

/T:[(][)] [=]xxx Time. Select files based on the relationship between their last update time and the time specified in the command. The /T: parameter may also contain an operator that reverses or modifies the meaning of the time specified, just like the date parameter.

Examples of /T:xxx Parameter

1. Process files updated before 11:20 a.m.
   `/T:(11:20a` or `/T:(11:20`
2. Process files updated before 11:20 p.m.
   `/T:(11:20p` or `/T:(23:20`
   –> military time
3. Process files updated after 11:20 p.m.
   `/T:)11:20p`

| | |
|---|---|
| | 4. Process files updated within 60 seconds after 11:21 p.m.<br>`/T:=11:21p`<br>5. Leading zeros in hour may be omitted.<br>`/T:)1:20` |
| /TF | Test First. Works similar to the /T parameter, but after test processing is complete, PCOPY will ask you if you want to execute the command as if the /T parameter were not present. If you respond <Y>, the command will be executed and files will actually be processed. |
| /U | Update files. Only copy files that do not exist on the target directory or duplicate filenames that are newer than their counterparts on the target directory or disk. Specifying this option results in the most current version of each selected file being in the target directory. |
| /V | Verify. Use the DOS VERIFY option while copying data. This causes all data to be reread after it is written to disk to ensure it was written properly. |
| /WH | WHy. Indicate why files are not selected for processing. Normally, PCOPY will not show filenames that are not selected for processing. Specifying /WH will cause PCOPY to display a message indicating why any file was not selected for processing. |
| /WT | Wait. Wait for disk to be mounted before beginning processing. This parameter is intended primarily for systems with floppy disks. Before beginning the copy process, PCOPY will ask you to place the proper disks in the drives. This allows you to have PCOPY on a different disk than the disks being processed. |
| /X | Move files. This parameter changes PCOPY's function to move files instead of copying them. After the file is copied, it will be deleted from the source location. If the file is moved to another directory on the same disk, the directory entry for the file is updated, but the actual data is not moved. This provides for the fastest move possible. |

## PCOPY Menu Operation

When you start PCOPY without command-line specifications, it displays a menu for entering copy specifications. Three menus are available for specification of options. You proceed forward and backward through the menus by pressing the <PgUp> and <PgDn> keys. Once you are completed specifying options, press <F10> to start processing. Press <ESCape> at any time to exit from PCOPY. If you need PCOPY help for batch operation, press <F1>.

You may save a particular combination of menu specifications for use at a later time by assigning copy profile names to them. PCOPY saves up to four different copy profiles. To use these in a subsequent execution of PCOPY, simply select one of the saved names from the menu. All options on the menus will be set to reflect the contents of the saved profile options.

Menu options are specified in one of two ways. First, you may be asked to key specific parameter information into parameter fields. Second, you select toggle options by positioning the highlight bars over options specified and pressing the <Enter> key. PCOPY will only allow compatible options to be specified, so turning on some options may automatically deactivate others. You see this happen when watching highlighted options.

Use the up- and down-arrow keys or the tab keys to move from option to option. A short help line is provided at the bottom of the display to assist you in understanding each option. It changes as you move the cursor from option to option on the display.

### First menu of PCOPY

- `Copy Profile`

  An area is provided for you to provide an optional copy profile name to be associated with these parameters. Provide a 1–8 character name. The parameters specified for this process will be saved and associated with this profile name.

To the right of the name field are names for up to four copy profiles previously saved. To use the parameters of any one of these profiles, press <1>, <2>, <3>, or <4>. The menu will automatically change to reflect the saved specifications.

- `From Path`

  Enter the file specification for files to be copied. The same rules apply as for command-line specifications. See that section of this document for details. Following are examples:

  ```
  C:\DIRNAME\*.EXE
  \DIRNAME\*.EXE
  \DIRname
  *.exe
  ```

- `Drives`

  Enter additional drives to be searched for the path specification in From Path above.

- `To Path`

  Specify the target location where files are to be copied. Following are examples:

  ```
  C:
  C:\DIRNAME
  C:\DIRNAME\FILENAME
  ```

- `Process Options`

  `COPY MOVE TEST VERIFY WRITES`

  Highlight the MOVE or COPY action to take. TEST mode causes PCOPY to execute without really copying data. Use TEST to check that your parameters are correct (/TF ,/X, /V).

- `Multiple Directories?`

  `ENTIRE DISK ALL SUBORDINATE DIRECTORIES`

  Indicate what directories are to be read to find files to be processed. SUBORDINATE DIRECTORIES indicates to read the current directory and all of its subordinate directories to find files to process (/S, /DS).

- `Directory Actions?`

  `CREATE NEW CREATE ALL CREATE/ADD LASTNODE MERGE SYNCHRONIZE`

  Specify actions to be taken at the directory level.

  CREATE NEW — Automatically create directories needed on the output disk (/DC).

  CREATE ALL — Automatically create directories for every directory read on the source disk (/DCA).

  LASTNODE — Use the last directory level from the source disk, and append it to the directory name on the target disk. This creates a new directory structure under the target directory specification (/DL).

  MERGE — Merge files from the source specification into the target directory. Delete old or exact duplicate files from the source location. Replace newer duplicate files on the target location (/ME).

- `File Filtering?`

  `NEW & MORE RECENT FILES ONLY NEW FILES PAUSE/ASK FOR EACH FILE MARKED FOR BACKUP READONLY HIDDEN SYSTEM`

  Indicate selection criteria for files to be processed:

  NEW — Copy files that do not exist on the target directory and files that exist but are newer.

  ONLY NEW — Process files that do not already exist on the target directory (/N).

  PAUSE/ASK — Stop and prompt for approval before processing each file (/P).

  MARKED BK — Process files that have been modified since they were previously backed up (/A).

  READONLY — Process files that have their readonly attribute set (/RO).

  HIDDEN — Process files that have their hidden attribute set (/SF).

  SYSTEM — Process files that have their system attribute set (/SF).

The second PCOPY menu. Access this menu by pressing <PgDn> from the first PCOPY menu.

## The Second Menu of PCOPY Options

- `LIST FILES NOT PROCESSED`

  Show filenames matching the filename criteria but were not processed for other reasons. Indicates filename and the criteria that caused it to be skipped (/WH).

- `SET DOS FILE CHANGED INDICATOR AFTER COPY`

  After copying the file, reset the DOS attribute indicating the file needs backup (/BA).

- `SET DOS FILE DATE TO TODAY AFTER COPY`

  After copying the file, set the date and time on the file to the current date (/RD).

- `FILL OUTPUT DISKS EFFICIENTLY WITH BEST FIT`

  Process files in a sequence that will allow them to efficiently fit on output diskettes. PCOPY will find files to completely fill a disk before going on to additional disks (/FI).

- `ERASE OUTPUT DISKS BEFORE USING ERASE ON ALL BUT FIRST`

  Erase all files on target disks before copying files to them. 'ALL BUT FIRST' is normally used during backup to add files to the first backup disk used (the last one used for the previous backup run), and then erase all subsequent disks before copying to them (/D- , /D2).

- `APPEND COPIED FILES TOGETHER IN ONE OUTPUT FILE`

  All files being copied are to be merged into one output file. The output file is the filename specified on the target specification (/AP).

- `FILES MAY BE DIVIDED INTO PARTS ACROSS DISKS`

  Split files onto multiple disks if there is not enough room on the target diskette for the entire file to fit. See the /SP command-line parameter.

- `ASK FOR MULTIPLE INPUT DISKETTES`

  Causes PCOPY to ask for more input diskettes after the first diskette is processed. Great when reading from a series of diskettes (/MU).

- `DISABLE SOUNDS DURING PROCESSING`

  Disables speaker beeps normally sounded to inform the user some action is needed (/SO).

## The Third and Last PCOPY Menu

- `Select files by size`

  If a numeric value is specified, only files larger than (>) or smaller than (<) the value are processed. The numeric value must be preceded by < or > (/SZ:).

- `Select by date From Date: to Date:`

  If one or more dates are specified, only files having a date less than or greater than the dates specified will be selected for processing (/D:).

- `Stop when Source Drives freespace >`

  If a numeric value is specified, PCOPY will check the source drive's freespace before copying each file. If the drive's freespace is greater than the value specified, processing will stop. Use with MOVE option to empty a disk up to a specified level of freespace (/SS:).

- `Stop when Target Drives freespace <`

  If a numeric value is specified, PCOPY will check the target drive's freespace before copying each file. If the freespace is less than the value specified, processing will stop. Use to copy files to a disk until it has less than n characters free (/ST:).

- `Duplicate Files: REPLACE SAVE OLD COPY ONLY DUPLICATES`

  Indicate how to process files that already exist on the target directory.

  REPLACE — Overlay files regardless of date/time (/R).

  SAVE OLD — Rename the target file and then copy the source file to the directory. The new name will have a number placed in the file extension to indicate the version of the file (/SA).

  ONLY DUPs — Select duplicate filenames for processing. Ignore all other files (/E).

- `Networks: NO FILE SHARING COPY FILE BEING READ COPY ALL IN USE`

  When using PCOPY in network environments, indicate how PCOPY is to proceed when files are found to be in use by other tasks in progress.

  | | |
  |---|---|
  | NO SHARE | Skip files in use (/SHN). |
  | READ | Process files being read by another task (/SHR). |
  | ALL | Continue to process any file already in use (/SHA). |

- `Process files in date sequence: ADCENDING DESCENDING`

  Process files in date sequence. Use to copy the most recent files, or the oldest files to a target location (/SN, /SO).

## Ways to Use PCOPY

1. Copy files from floppy disk to hard disk

   PCOPY is safer than DOS COPY when copying files to your system. It prevents you from accidently replacing a file with another of the same name.

   When PCOPY encounters a duplicate file, it will stop and ask if the file is to overlay the already existing one. You can tell PCOPY to REPLACE or UPDATE files with /R and /U command-line parameters.

   Following are samples of the simplest copy commands using PCOPY:

   If you are copying to the current directory on the hard disk:

   ```
   PCOPY A:*.*
   ```

   To copy to a specific directory:

   ```
   PCOPY A:*.* \SPECDIR
   ```

2. Taking changed files from work to home

   It is common to need to take files from home to work or vise versa. Since it is difficult to keep track of all the files you may have updated, it is convenient to let PCOPY find the updated files by copying all files updated within the last *n* days to a diskette. You reduce copy time by not copying an entire directory and are sure to get all newly changed files.

   ```
   PCOPY *.* A:/d:3 <== updated last
               3 days
   PCOPY *.* A:/D:)1DEC90 <== After
               date
   ```

3. Copy files modified before/after a time

   Sometimes the only way to identify recently changed files is by using the time they were updated. PCOPY can select files by time of day. You might use this option to determine what files were updated after 5:00 p.m. Remember, with the /T TEST option, PCOPY serves as a unique facility to locate and list files matching your selection criteria. PCOPY's unique selection facilities can be used to locate and list files. The /T option indicates to scan disks and simulate processing but do not actually move or change any data.

4. Quick move files to a new directory on the same hard disk

   Moving files with DOS requires two commands: COPY the data and then DEL the files from their original location. In addition to requiring two commands, it forces DOS to copy the data. If the file is larger than the available freespace on the hard disk, a copy of the data is not possible.

   PCOPY moves files to new directories on the same disk without having to physically copy the data. It simply moves the file reference to a new directory. Even huge files move in an instant!

   ```
   PCOPY \OLDDIR\*.DAT \NEWDIR /X
   ```

5. Copy only duplicate files

   To update a directory with duplicate files from another directory or disk:

   ```
   PCOPY \FROMDIR1 A:*.* /E
   ```

6. Copy multiple groups of files by extension names

   Use PCOPY to process multiple file selection specifications with a single copy command. Separate each with a comma.

   ```
   PCOPY *.BAT,*.EXE,*.COM \EXEDIR
   ```

7. Copy files excluding files of one or more extension names

   The OMIT feature lets you exclude files from processing using command-line specifications. It is commonly used to exclude program files when copying to diskettes.

```
PCOPY *.* A: /O:*.EXE,*.BAT,*.COM
```

8. Copy files containing specific characters in their name

   When you need to select or find files having a specific word in their names, but the filenames do not begin with the word, PCOPY can locate them.

   Copy all files having names containing the 3 consecutive characters **ZIP**.

   ```
   PCOPY *ZIP*.* \ZIPFILES
   ```

9. Display files having specific name

   Use PCOPY to find all occurrences of files by name on a disk. PCOPY can scan a disk and display or process files from multiple directories. The /S (SCAN) parameter causes PCOPY to search all directories.

   ```
   PCOPY *.DOC \TEST /S/T
   ```

10. Copy only newer files between disks and directories

    One of the most common uses of PCOPY is to copy data between diskettes and hard disks. Use the update (/U) parameter to make sure only newer files are updated. Add the "WHY" (/WH) parameter to cause PCOPY to show any files that were not processed. Note that you can specify the target directory name without trailing \.

    ```
    PCOPY A:*.* C:\NEWDIR /U/WH
    ```

11. Copy a disk including all directories to a new disk

    PCOPY copies directories too. Simply specify the Directory Copy parameter (/DC). The file will be copied to the same directory name on the target drive. If the directory does not exist, it will automatically be created for you. Adding the Scan (/S) parameter causes PCOPY to scan all disks on the source drive. Using a combination of /S/DC, you can copy all files and directories from one disk to another.

    ***Note:*** /DC only copies directories when files are to be copied into them. This ensures that unneeded directories are not also copied. To copy every directory even if no files are copied, specify the All Directory Copy (/DCA) parameter instead of /DC.

    ```
    PCOPY A:*.* F: /DC/S
    ```

12. Copy a directory and all subordinate directories

    To copy or move files from a directory and all its subordinate directories use PCOPY's /DS parameter. The following example scans the directory C:\WP and any subordinate directories to copy all files having the extension name .TOM to a diskette.

    ```
    PCOPY \WP\*.TOM A: /DS
    ```

13. Fill one or more diskettes from hard disk files

    DOS COPY and XCOPY are unable to copy a set of files to multiple diskettes. When the first diskette is full, they stop.

    PCOPY is not limited to copying to a single diskette. It detects when a disk is full and asks if the copy is to continue on to additional output diskettes. PCOPY also makes maximum use of diskette space.

    When you are attempting to copy many files to diskettes for backup or transporting, you want to use as few diskettes as possible. The PCOPY Fill (/FI) parameter causes PCOPY to fill output diskettes without wasting space. With the /FI parameter, when PCOPY is unable to fit a file on an output diskette, it saves that filename for later processing and searches for another file that will fit on the disk. This results in significant savings in diskettes and time to process them.

    ```
    PCOPY C:\WP\*.* A: /FI
    ```

14. Copy files larger than a floppy disk to multiple diskettes

    Unlike DOS COPY and XCOPY, PCOPY can copy files larger than a diskette to multiple diskettes. This is accomplished by splitting large files into multiple parts. PCOPY only splits files when there is insufficient space on the output disk and when the split parameter is specified. The Split (/SP) parameter tells PCOPY it is OK to split large files. Following is an example:

    ```
    PCOPY C:\WP\*.* A: /SP/FI
    ```

    To restore split files, simply use PCOPY to copy the files back to a disk large enough to contain the entire file. PCOPY will find the split files and automatically recombine them.

```
PCOPY A:*.* C:\WP /MU
```

15. Backup from hard disk to floppy disks

PCOPY is not a full backup system, but its features allow it to be used for backup processing. It is very effective for backing up one hard disk to another. Following is a basic explanation of backup procedures.

The most common backup technique is referred to as Incremental Backup. This technique uses two different backup commands to implement initial full backup and daily incremental backup.

Incremental backup lets you avoid repeatedly backing up your entire system each time you want to back up. Instead, the first time you backup copy your entire system to disks. Then, each subsequent day (or other backup period) you only back up the files that have changed since the previous back up. If you are like most users, your "daily" backups will require only a couple of minutes because you only modified a small portion of your files since you last backed up.

A special DOS maintained file archive attribute supports incremental backup. The flag is an indicator field that is kept by DOS in each file's directory entry. It is stored along with file date, time, and size information. Whenever DOS, or an application, creates or changes a file, this archive attribute flag is set to indicate the file needs to be backed up. Programs that support backup use this flag to distinguish between files that need backup and those that have already been backed up.

Whenever a program backs up a file, it should reset the flag to indicate backup has been done. Then, incremental backup processing will know to skip the file and not back it up again until it has changed.

## Full backup

Following is an example of using PCOPY to backup hard disk C: to diskettes in drive A:.

First, we will show the PCOPY command to make the first backup.

```
PCOPY C:*.* A: /DC/S/D-/A/LG/FI
```

The above command causes PCOPY to copy all files on drive C: to drive A:. /DC and /S cause directory names to be copied to the target drive and make PCOPY back up from all directories. Since this is the first backup, /D- is included to force all files on drive A: to be deleted prior to the copy. /A causes PCOPY to reset the DOS archive flag on each processed file to show the files were backed up. The /LG parameter asks PCOPY to write a log record to PCOPY.LOG for each file written. The log file can be used to locate the diskette containing a file to be restored. /FI causes PCOPY to fill all disks efficiently and make optimum use of diskette space.

This command should only be run for the first backup.

```
INCREMENTAL BACKUP (Daily backup)
```

The following command causes PCOPY to scan your disk to find files that have changed since your previous backup, and add them to your already started set of backup diskettes.

```
PCOPY C:*.* A: /DC/S/D2/A/LG/FI/B
```

The incremental backup command is almost the same as the full backup except for two parameters. The /B parameter causes PCOPY to only process files that have the archive (backup) attribute set. Processing will be confined to changed files. /D2 is used in place of /D-. /D2 indicates that PCOPY is to append files to the first backup disk and then to erase all files before reusing any additional diskettes.

After running your full backup, keep track of the last diskette written to. This will be the first disk provided for the next Incremental Backup. Label a set of backup disks and number them sequentially. Use a specially marked diskette sleeve for storing the last disk used. This will make it easy to identify when running backup again.

## Restoring files from backup disks

Use PCOPY to restore any file from the backup disks. The PCOPY.LOG file contains a complete list of all backups done. You can examine this file to determine if and when a file was backed up. Then use the following PCOPY command to restore any files.

```
PCOPY A:filename.ext C: /MU/DC/S
```

16. Combine one or more files to a single new file

Use PCOPY to combine two DOS text files into a single output file. The following command copies two files to a single file, DOS.TXT. The append parameter causes them to be concatenated together.

```
PCOPY DOC.TX1,DOS.TX2 DOS.TXT /AP
```

17. Copy and save duplicate files with a new generated name

Use PCOPY when you need to copy updated files to diskettes that already contain previous versions of the same files. The /SA parameter indicates PCOPY is to save old versions of files by renaming them with version numbers in their extension names. PCOPY.C would be renamed to PCOPY.C01 before a new PCOPY.C is copied to the same diskette.

```
PCOPY \UPDATES \CSOURCE /R/SA
```

18. Merge two directories

Use PCOPY to move all files from one directory to another making sure the most current version of files in both directories are kept.

Merging files from one directory into another involves three actions. First, PCOPY moves files from the source directory that do not already exist on the target directory. Second, it moves newer duplicate files to the target directory. Third, it deletes files from the source directory that were not copied to the target directory.

```
PCOPY \fromdir \todir /ME
```

19. Execute any command with selected files as parameters

Since PCOPY has file selection capabilities not available in other programs, it can be used to select files to be processed by other commands and applications. For example, the following command will start an EDIT command for all P*.C files found by PCOPY. With the /CD parameter, PCOPY only starts other commands. It does not copy or move data.

```
PCOPY P*.C /CD:EDIT[#
```

20. Search multiple directories for files to copy

Use PCOPY to scan a set of directories and copy files to a new location. This is useful if you are not sure of exactly what directory a filename exists in or if files from multiple directories are to be copied. Use the /S or /DS parameter.

Here is a sample command:

```
PCOPY LOOKFOR.ZIP A: /S/DR:CDE
```

21. Delete files on target disk and copy files to it

You may keep a work diskette handy by your machine to use to transport files between home and office. Use the /D- command with PCOPY to erase any old files on the disk before copying new files to it. This saves you from having to use a delete command and press Y to delete all files. The following command will erase the disk in drive A: and then copy files to it.

```
PCOPY PW*.C A:/D-
```

22. Restarting a previously incomplete copy

Sometimes it is necessary to stop a copy and start again at a later time. Even with a single diskette, the DOS COPY command is unable to resume processing from a specific filename. The PCOPY /F:filename.ext parameter provides PCOPY with a starting filename. The following command copies all files from PCOPY.C in a directory to diskette drive A:

```
PCOPY *.* A: /F:PCOPY.C
```

If you are referring to a hard disk, you do not need the /F command to resume processing. Simply specify the /U parameter. PCOPY will skip all files previously copied and start when the first uncopied file is reached.

23. Freeing disk space by copying files to archive disks

When your hard disk becomes full, you can use PCOPY to locate old files and move them to diskettes for archival storage. The following command finds files in the ARCHIVE directory that are older than a specified date and move them to diskette. Include the /SS:nnn command to stop processing when sufficient freespace is available on the hard disk.

```
PCOPY \ARCHIVE A: /X /D:(1jan89
/ss:1000000
```

24. Transporting an entire disk to another computer
Copying one disk and all its subdirectories to another disk is easy. The first command copies all files to diskettes. The second command copies the files and directories from the diskettes back to any hard disk.

```
PCOPY C:*.* A: /DC /S /FI
PCOPY A:*.* C: /DC /S
```

25. Determine differences between files in two disks
One of the most useful functions of PCOPY does not copy files at all. To compare and report the differences between files on a diskette and the files on a hard disk directory, use the PCOPY command in test mode.

```
PCOPY A:*.* \DIRNAME /U/WH/T
```

The /T parameter indicates test mode no copy or move is to be done. The /U indicates to update any newer files. The /WH indicates to show why any files would not be copied. The resulting display shows the new files, which are newer versions, and what files are already on the drive as duplicates. It will also indicate what files are older copies of the same filenames.

26. Keeping files on two computers in synchronization
The /DY synchronization parameter indicates PCOPY is to make the target directory contain the same files as the source specification. It saves time by only copying files needed. An important feature is its ability to remove any files from the target drive that are not on the source drive being copied from.

This command is especially useful for transporting files from work to home and home to work using diskettes. A quick command at the end of each day will copy the files and remove deleted ones.

```
PCOPY \WORDPROC\*.* A: /DY
  Work to diskette
PCOPY A:*.* C:\WORDPROC /DY
  disk to home
```

27. Saving generation backup copies of files
Products that back up files before changing them often do not consider situations when a user stops and starts an application many times between backups. Unless multiple backup files are maintained, good versions of a file can be lost when good backup versions are replaced with bad versions.

Since PCOPY can create backup files with consecutively numbered names, you never lose a backup version. Just use PCOPY to make a copy of the file(s) to a backup directory before starting your application.

```
PCOPY %1 \BACKUP/SA/NW/U
  Make backup
123 %1 <=== Start application
```

After you next back up your system, delete the PCOPY backup versions to free space on your hard disk. Use the PDEL delete utility to delete all files in the BACKUP directory without being prompted to confirm the delete request.

After Backup:

PDEL \BACKUP*.*/NC Erase backup directory

28. Quickly copying files from one diskette to another
Since multiple floppy drives on a system cannot be accessed at the same time, it is very inefficient to copy files from one floppy disk to another. Instead, create a temporary directory on your hard disk, and first copy the files to the hard disk directory. Then move them from the hard disk directory to the output diskette. This greatly reduces copy time by avoiding multiple stopping and starting of floppy drives. If you can use a RAM disk as an intermediate drive, the copy will be very fast.

```
MD C:\TEMP
PCOPY A:*.*/NW
PCOPY C:\TEMP B:/X/NW
RD C:\TEMP
```

# PDAILY

**Copyright © 1986 by Norm Patriquin**
**Version 1.0**

## What Is PDAILY?

PDAILY allows the user to execute a set of DOS commands on a once per day basis. It is most useful for customizing your daily system startup procedure to perform some selected functions the first time the system is started each day and to bypass them if the system is started again the same day.

The system was developed to remind me of my daily planning tasks once each day. PDAILY works in conjunction with the DOS ERRORLEVEL batch command facility. The following statements show how selected commands are executed once per day.

**Example:**

```
ECHO OFF
PDAILY
IF NOT ERRORLEVEL GOTO NOTNOW
PLANNER
DEL *.BAK
BACKUP *.* A:/A
:NOTNOW
MENU
```

## Command Parameters

PDAILY allows several command parameter switches to be specified to customize the process. All of these parameters are optional. The command format and switch parameters are described below. A quick review of command parameters is always available by entering the PDAILY command with the /H parameter.

**Command Format:** PDAILY [logfilename] (/ switches)

| | |
|---|---|
| *logfilename* | You may optionally specify the alternate name of a log file PDAILY is to use. This is a zero-length file that is used to keep the current PDAILY status. This feature is useful to keep PDAILY status for more than one event or activity. Use a different logfilename for each PDAILY event controlled. If not specified, PDAILY.DAT is used. |
| /H | Display help about PDAILY. |
| /N | Return the proper ERRORLEVEL command, but do not change it. Running PDAILY again will produce the same ERRORLEVEL the next time it is run, even the same day. |
| /R | Reset PDAILY status from ALREADY DONE TODAY to NOT YET DONE TODAY. The next time PDAILY is run, it will indicate the commands are to be executed again. |
| /T | Print status messages to indicate what ERRORLEVEL PDAILY has set. |

# PDELETE

**Version 4.4**
**Copyright © 1992 by Norm Patriquin**

## What Is PDELETE?

PDELETE is an advanced Delete command to be used in DOS operating systems. It may be used in place of the DOS Erase and Delete command or instead of it. Like other Patriquin utilities, PDELETE gives you more control of command processing and provides a much safer way to process files. PDELETE also provides several features not found with the standard DOS command.

Unlike other delete utilities, PDELETE does support DOS paths and wildcards. It keeps you aware of what files are being processed and provides an emergency stop feature just in case you have started something you feel is wrong.

PDELETE's file selection parameters provide you with numerous options for specifying the files to be deleted. You may select by directory, file, date, attribute, size, and even starting and ending filename.

The PDEL.DOC file contains a section outlining all changes for all releases of PDEL. If you already have a current release, this file will let you identify the changes since the version you have.

This archive should contain the following files:

| | |
|---|---|
| PDEL.EXE | PDELETE execution file. |
| PDEL.DOC | PDEL documentation file. Use the following command to print PDEL documentation.<br>`COPY PDEL.DOC PRN:` |
| PDREADME.TXT | A file to explain the contents of PDEL system. |
| WHATSNEW.DEL | Enhancement notes for this release. |
| ORDERFRM.TXT | Text form for ordering PDEL. Use the following command to print this file.<br>`COPY ORDERFRM.DOC PRN:` |
| PDEL.1 | A one-line description of PDELETE. |
| PDEL.DES | A multi-line description of PDELETE. May be used when uploading PDELETE to BBS systems. |

## Features

- Advanced file selection.
- Emergency STOP by pressing any key.
- Delete entire directories and remove them from disk.
- Optionally wipe file data from disk during delete to prevent it from being recovered by unauthorized person.
- Optional verify feature lets you approve each file before it is deleted.
- Optionally deletes HIDDEN / SYSTEM / READONLY files.
- Delete files until specified freespace is available.
- Advanced DOS Pattern capability. *TEST*.*
- Test feature allows you to try a command before actually deleting any files.
- Deletes from multiple disk drives with a single command.
- Accepts a list of files to be deleted.
- Exclusion capability allows files to be protected from deletion.

## Command Parameters

PDELETE allows several command parameter switches to be specified to customize the process. All these parameters are optional. The command format and switch parameters are described below. A quick review of command parameters is always available by entering the PDEL command with no parameters.

**Command Format:**

| | |
|---|---|
| PDEL | filespec,filespec,filespec<br>[ (/ switches) ] |
| *filespec* | Enter a DOS file specification. Pathnames and standard DOS wildcard characters are allowed.<br>Multiple file specifications may be provided as shown in the above example. Separate each filename pattern with a single comma. For example:<br>`PDEL *.BAK,*.BK!,*.TMP`<br>Special pattern selection like *DISK*.* may be specified. *DISK*.* would select all files that have the chars 'DISK' anywhere in the first portion of the filename. *DI*S*K*.* type specification also works.<br>As an alternative, a file containing a list of input files may be supplied. Create a file containing lines with a filename or complete pathname on each line. Provide this filename as the source filename to PDEL and add an @ character before the name. The list of files will be processed as if they were all specified on the input command line. When the /S parameter is specified, an input file list may only contain file |

names. It may not contain directory names.

For file list processing, PDEL understands the output of the DOS DIR command. You may use the DOS redirection feature to direct DIR output to a file, edit the file to add or remove entries, and then provide this to PCOPY as input. For example:

```
DIR A: >DIRLIST
(edit dirlist)
PDEL @DIRLIST B:
```

| | |
|---|---|
| /A | Archive — Selects only files that have been marked updated by DOS. This flag is updated by DOS whenever a file is processed with an intent to change or create new data. |
| /BA | Batch file processing. PDEL always checks to determine if the user has pressed a key to pause processing. This check can interfere with programs that store keystrokes in the keyboard buffer for subsequent processing. The /BA parameter sets no-windows mode and disables the key press check to leave stored keystrokes undisturbed. |
| /BW | Forces PDEL to use monochrome colors even when it detects color display environment. (Black & White) |
| /C | Overrides /NC when /NC is placed in the DOS environment variable. /C indicates to ask before deleting any files if a file pattern of *.* is specified. |
| /C2 /C3 | Specify /C2 or /C3 to display the window processing screen in an alternative color scheme. To make this color change permanent, specify the parameter in the DOS environment variable. |
| /CE | Clean Empty directories. Usage: PDEL <path> /DE. PDEL will scan the entire disk and remove all empty directories in the specified tree. To process an entire drive, specify the root directory. Ex. PDEL C:\ /DE. ***Note:*** Only /T can be used with this parameter. No files will be deleted at any time. |
| /DE | Delete Empty directories. Using this parameter, PDEL will remove all directories in which all files have been deleted from. For example, if PDEL deletes all files from the directory C:\WORK, it will then remove the directory. |
| /D:[(]xxx Date. | Select files based on the relationship between [)] their last update date and the date specified. This [=] parameter includes a date or number of days value. It may also contain a modifier that reverses or modifies the use of the date specified.<br>When xxx contains a date, the date is compared to the last update date of the file. The last update date of the file is the date contained in its directory entry. The file is selected if the condition specified by the condition modifier is met. The condition modifier meanings are<br>'(' = Select files whose date is older than the date specified.<br>')' = Select files whose date is newer than the date specified.<br>'=' = Select files whose date is equal to the date specified. If the condition modifier is not specified, ')' is assumed.<br>Date values may be specified in any of the following formats:<br>MM/DD/YY 12/31/80<br>MM-DD-YY 12-31-80<br>DDMMMYY 31DEC80<br>If xxx contains a numeric value, that positive number is used to calculate a date value a number of days prior to the current date. Then that date is used as in the calculation above. |

***Example:***

| | |
|---|---|
| /D:(1JAN86 | Process files updated before 1JAN86 |
| /D:)1JAN86 | Process files updated after 1JAN86 |
| /D:=1JAN86 | Process files updated 1JAN86 |
| /D:(5 | Process files modified prior to 5 days ago |
| /D:)5 | Process files modified within the last 5 days |
| /D:=5 | Process files modified 5 days ago |

/DI Delete the disk DIrectory specified in the command. This will remove the directory, all files within it, and all the directories and files subordinate to it.

Only use this as a stand-alone parameter. Use a command like the following one to remove a directory from your disk:

```
PDEL \OLDDIR /DI
PDEL *.*/DI
 – or –
PDEL \OLDDIR\*.* /DI
```

/DR:xxx Use to process multiple disk drives. Processing will attempt to delete the specified files from all drive letters provided with the /DR:xx parameter.

For example, the following command will delete *.BAK files from hard disks C:, D:, and E:.

```
PDEL *.BAK/DR:CDE
```

/EQ Exit PDEL without user intervention. Before PDEL exits window operation, it requests you to press any key. This parameter bypasses this request so you can use PDEL in DOS batch files.

/EX Exclude files. The exclude facility allows you to provide a list of pathname and filename specifications that are to be excluded from processing.

The exclude facility is implemented with two actions. First, you will need to create an exclude specification file and place it in the same directory where PPDEL.EXE is installed on your hard disk. This file will contain the names of all the files and paths to be excluded. Name the file PDEL.EXC.

You may specify DOS pattern names to exclude groups of files with a single specification. Like DOS, use the '*' to indicate the last part of a filename or filename extension may be any set of characters. Also like DOS, use the '?' to indicate that any single position of a filename may be any character. In addition to DOS, two additional pattern features are available. You may place an '*' anywhere in the filename to indicate any set of characters may exist in this point in the name. For pathnames, you may place an '**' at the end of a directory path specification to indicate the specification applies to the specified directory and all subordinate directories. Using '**', you can exclude an entire path with a single specification. Following are examples:

1. *.EXE
2. *.COM
3. TEST*.*
4. *TEST*.*
5. DO?FIX.*
6. PC*XX.*
7. C:\DOS\*.*
8. \TEST*.*
9. D:\GAMES**\*.*

Explanations for the previous examples:

1. Excludes all filenames with the extension .EXE from processing.

2. Excludes all filenames ending with extension .COM.
3. Excludes all filenames beginning with the letters "TEST" from processing.
4. Excludes all filenames having the characters "TEST" anywhere in the first portion of the filename.

5 Exclude files having 'DO' as the first 2 characters of the filename, and 'FIX' as the 4th, 5th, and 6th filename characters. The 3rd character of the filename may be any character.

6. Exclude files having a name beginning with the 2 characters PC and where the first portion of the filename ends with the characters 'XX.' Any 0 to 4 characters may exist between the 'PC' and 'XX' in the first portion of the filename.
7. Exclude all files from the DOS directory on drive C:.
8. Exclude all files from the TEST directory existing on any drive searched.
9. Exclude all files from the D:\GAMES\ directory and also exclude all files and directories subordinate to the GAMES directory.

Activating exclude —

Exclude will only be functional when the /EX command parameter is specified. When /EX is specified, you must have a valid exclude file. PDEL will not continue without one. If you wish exclude to be active during all execution, specify /EX as part of the PDEL= DOS environment variable. See later in the documentation for a description of this feature.

| | |
|---|---|
| /F:file | First. Indicates that processing is not to start until the specified full filename is encountered in the scan of the directory. |
| /FS:nnnn | Freespace. PDELETE will delete files as long as there is less than nnnn thousand (nnnn * 000) of free space available on the disk. This is very helpful for cleaning up disks but retaining as many files as possible. |
| /L:file | Last. Indicates that processing is to stop after the specified full filename is encountered in the scan of the directory. The file specified will be processed if it matches selection criteria. |
| /M | Stop screen from scrolling from scrolling past the end without user intervention (MORE). |
| /NC | PDELETE asks "Are you Sure?" when you request files to be deleted with "*.*" patterns. To ignore this question, specify /NC. Place this in the DOS environment variable with SET PDEL=/NC to prevent PDEL from ever asking if you are sure. |
| /NT | Prevent ending statistics about freespace and files deleting from printing. |
| /NS | Sets all sounds generated to the speaker OFF. (No Sound) |
| /NW | No Windows — Do not use windows during processing. All screen output will be done through standard DOS facilities. |
| /O: filename-list | Specify a list of filename patterns that are to be Omitted (excluded) from PDEL processing. If multiple filename patterns are specified, separate them with commas.<br>For example:<br>/O:\*.BAK,\*.EXE,TEST*.*<br>An alternate form of the /O parameter allows all files but the original filename pattern to be deleted. Specify /O without any filename. For example, to delete all files except *.EXE files ===> PDEL *.EXE /O |

| | |
|---|---|
| /P | Pause before each file is updated to allow user to approve or reject the action for each file. |
| /RE | Use the /RE parameter to indicate any file with special attributes is also to be removed from the disk. Files with HIDDEN, SYSTEM, or READONLY attributes will be processed. |
| /RO | Include files marked with DOS readonly attribute. |
| /S | System — If a source directory is supplied, that directory and all subordinate directories will be searched for files to delete. If no source directory is specified, all directories on the disk will be searched for the specified filename pattern to delete. |
| /SH | Include files marked with DOS Hidden attribute. Normally, PDELETE will bypass these files. |
| /SF | Include files marked with DOS System attribute. |
| /SO | Process files from oldest to newest. |
| /SN | Process files from newest to oldest. |
| /SZ: [(] [)] nnn | Select a file based on its size. The left paren indicates that files less than the indicated size are to be selected. The right paren indicates to select files greater than the size. If right or left paren is not specified, a left paren is assumed. Following are examples.<br>/SZ:100 Select files less than 100 characters<br>/SZ:(100 Select files less than 100 characters<br>/SZ:)100 Select files greater than 100 characters |
| /T | Test the command. Use this switch when you would like to test the command entered without actually updating any files. This is one of the best features of PDEL. Since PDEL is so powerful, the /T parameter is provided to allow you try any command in a simulate mode before actually deleting any files from your disk. Provide the /T parameter with any PDEL command to get a trial run without having any files deleted.<br>For example: PDEL *.BAK/T/S will search all directories in the current disk for *.BAK filers and will show what files would be deleted if this same command was run without the /T parameter. |
| /TF | Like test, but after test operation is complete, you will be asked if delete is actually to be done. (Test First) The test first allows you to try a command and then immediately execute the command again without the test option. It is a way to perform the test and live runs using a single command. |
| /W | Forces window operation. Since PDEL defaults to window operation, this parameter is only useful when you have previously deactivated window operation by placing /NW in the PDEL= DOS environment variable. |
| /WI | For data security, erase all disk areas that contain the file when it is deleted. This will prevent unauthorized users from recovering the file or examining its data by using special disk utilities. |
| /WH | PDELETE will indicate to you WHy files did not get processed. If a file is bypassed, normally PDELETE will not tell you why or when.<br>Use /WH when you want to determine why PDEL has not deleted a file you felt it should. /WH causes DEL to show why it bypassed processing for any files.<br>Setting of default parameters (DOS Environment) Default parameters are parameters that do not need to be specified each time PDEL is started. Any PCOPY parameter may be specified as a default parameter. |

Use the DOS SET command to specify the parameter name and contents to be used by PDEL each time it is started. The following example indicates that PDEL is always to assume /NC and /NW are specified:

```
SET PDEL=/NC/NW
```

Once this DOS command is issued, each subsequent use of PDEL will operate as if these parameters have been specified on the command line. The SET command is most commonly placed in the AUTOEXEC.BAT file so it may be in effect every time your machine is started.

***Examples:***

Delete all *.BAK files on current default drive

```
PDEL *.BAK /S
```

Delete all *.BAK files over 30 days old

```
PDEL *.BAK /S/D:(30
```

Remove directory \ACCTING1 from drive A:

```
PDEL A:\ACCTING1 /DI
```

## Patriquin's Hard Disk Utilities Registration Form

Name: ______________________________

Company: ______________________________

Address: ______________________________

City: ______________________________

State: __________ Country: ______________

Zip: __________ Phone: ______________

**Remit to:**

Patri-Soft
5225 Canyon Crest Dr, Suite 71-358
Riverside CA, 92507

(714) 352-2820 (Voice)
(714) 352-2825 (Data)
(714) 352-1527 (Fax)

Visa/Master Card orders are also accepted through telephone orders. Call Public Software Library: 800-242-4775 or 713-665-7017 to order.

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Payment by:

☐ Check or ☐ Money Order
(made out to Patri-Soft)
☐ Visa ☐ Master Card

Authorized Signature: ______________________

Expiration Date: ______________________

Card Number: ______________________

☐ Registration of all Patriquin Utilities with program disk and User's Guide.
each ........................................$45.00 $______

☐ Update Service — in addition to above, send the next version automatically when it is available.
each ........................................$25.00 $______

☐ PCOPY — File Copy, Move, Backup, Update, more!
each ........................................ $20.00 $______

☐ PSEARCH — Locate files on your disk.
each ........................................$20.00 $______

☐ PDELETE — Save space with automatic disk cleanup
each ........................................$20.00 $______

☐ PPRINT — Advanced file print w/special formats
each ........................................$20.00 $______

☐ PALERT — Warns you of low disk space
each ........................................$15.00 $______

☐ PATTR — Manage attributes of DOS files
each ........................................$10.00 $______

☐ PDAILY — Automatically execute commands once/day
each ........................................$10.00 $______

☐ Update a registered set with new disk and manual — show proof of previous registration SER=__________
each ........................................$15.00 $______

SHIPPING/HANDLING FOR ALL ORDERS
ADD ........................................$5.00 $______

California residents, add state sales tax
........................................7.75% $______

TOTAL........................................ $______

# PocketD

**(D.EXE) Directory Listing File & Management Program**
**Version 1.67c**
**Copyright © By Jeff Rollason, PocketWare 1990–1992**

## Introduction

PocketD is a small, flexible, powerful color directory listing program and exceptionally powerful DOS utility. Its highly generalized design and range of built-in features allows it to be used for almost any purpose, including for example: virus checking, disk cataloging, disk backup reminder, reversing unintended copies, DOS menu system, programmer's MAKE system, and much, much more.... PocketD is the one program most likely to fulfill all of your file management needs and probably the nearest you will get to the DOS user's equivalent of the Swiss Army knife.

## Features

PocketD is 25K in size and supports over 120 options, including the following key features (expanded and detailed in the file FEATURE.CMP):

- Full ON-LINE HELP giving direct access to help on each option
- Optimized file COPYing to multiple disks
- Safe file/dir COPY/MOVE/RENAME selected by any criteria
- Fast MOVE by copying directory entry, rather than COPY+DELETE
- SEARCH for files across multiple drives and subdirs
- Search or view contents of ZIP, ZOO, LZH, ARC, PAK, PKA, & ARJ archives
- Advanced wildcard system, allowing multiple filespecs such as "*TMP*F*" or "*=D*" (match any name containing a digit 0-9) etc.
- Highly programmable execution and display TEMPLATE(s), providing a viable alternative to a command-line programming language, allowing the creation of powerful interactive utilities and the emulation of programs such as WHICH, NCD, & MAKE
- Detection of, and testing for, PKLITE, DIET, and LZEXE compression
- TOUCH and full ATTRIB for modifying file and dir dates/attributes
- On-line BROWSE of file lists for VIEWING, DELETION, RUNNING, etc.
- A user-defined color MENU system for use in BAT files
- Analysis of ENVIRONMENT variables (e.g., PATH LIB, INCLUDE, etc.), STRING SEARCH in binary or text files for presence or absence of literal or "typed" strings (e.g., alphabetic, non-numeric, etc.)
- Automatic BEST-FIT directory listing format option
- RECURSED SUMMARY of subdir contents within directory listing
- BAR-CHART display of relative file/subdir sizes
- Display of EXECUTION SIZE of programs
- SORT by 6 keys (e.g., type, size, date) and by row or column
- Full COLOR options, allowing assignment for specified filespecs
- Automatic support for 80x43, 132x60, and any other screen sizes
- 37 user-defined parameters

## Overview

PocketD is good for both the occasional and the advanced user. It provides the usual day-to-day file operations, such as file copying, in a safe and easy-to-use format. It also bends over backwards to accommodate each user's particular tastes by offering vast numbers of tailoring options.

PocketD's multiple features can be combined to provide a powerful and highly generalized DOS tool. The following gives a couple of examples:

```
D /WTc A:
```

This compact example will search the whole disk (W) for files modified today (T) and copy them (c) to A:. This also gives the user the opportunity to view each file before copying it. Extra conditions can also be added:

```
D /WTc A: /E{20k!'Microsoft'
```

which will only include executable files (E) less than or equal to 20 kilobytes (20K) and excluding those that contain the string 'Microsoft' ("!" specifies "NOT").

```
D /WR .GIF [GIFVIEW $w] [COPY $w A:
// PKZIP SAVE $w]
```

This example searches the whole disk for graphic *.GIF files. The user can then browse these and optionally choose to view particular files using the program GIFVIEW and/or optionally copy them to drive A: and add them to the ZIP file SAVE.ZIP. The user can process any file in any order and any number of times.

## Doubts

Q1. Why use a program like PocketD when there are so many attractive alternatives, such as PCTOOLS or NORTON Commander which provide menu-driven interfaces to file management?

A1. The reason is that these menu-oriented packages provide very good point-and-shoot interfaces to file VIEW, COPY, MOVE, and DELETE, but do not allow the user to construct automated file management procedures, such as:

"Take all the ZIP files on drive A: and unpack each into its own subdirectory on drive C:"

This is not possible with these programs without repetitive keywork. Such operations require the use of packages such as MKS Shell or Command Plus and custom written routines to automate such operations.

PocketD can perform operations of this type conveniently and efficiently from a single command line, allowing the user to preview the operation before running it (e.g., from drive C:)

```
D A:.ZIP [MD $n//PKUNZIP $w
   $n\]? Preview and..

D A:.ZIP [MD $n//PKUNZIP $w
   $n\]R Run
```

Q2. Why should I use PocketD when there are other existing rivals?

A2. The only obvious rival is the much larger DX program by Econo-soft. DX has specialized features that PocketD does not support, such as "U.S. Defense standard" file deletion, file compare, and the ability to touch or rename files inside archives.

Despite its smaller size, PocketD still has more options, including advanced string searching inside files, a built-in file viewer, and full on-line help, not supported by DX. It also has a much more powerful template facility and much more flexible directory display (see the file FEATURE.CMP).

Q3. Why use PocketD when I do not require complex file management?

A3. PocketD is an attractive full-color directory listing program with many handy features such as automatic display format selection, which chooses a listing format for you. It also has a much more agreeable and powerful COPY facility than DOS's COPY, providing the user protection against common errors and giving informative feedback about the copy/move operation being completed.

PocketD makes day-to-day operations more pleasant and less error-prone (e.g., backing up of the day's files to disk). It allows the user to discard many utilities that are already used. PocketD saves disk space and provides all of these facilities with a single common range of command options.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the PocketD files in the directory \SECRETS\POCKETD. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the PocketD files to a directory that is included in the search path.

The following is a description of the contents for each file included with PocketD.

### Main files

| | |
|---|---|
| README.D | Read this file first. |
| D.EXE | The program PocketD (26K) help files that follow are optional. |
| D.HLP | The one page help (2K) view by **D /?** |

| | |
|---|---|
| D.MAN | On-line option manual (55K) view by **D /?x** |
| D.GDE | On-line User Guide (39K) view by **D /??** |
| D.DOC | On-line General Doc (23K) view by **D /?doc** |
| EPSON2A.COM | A program to convert the file FEATURE.CMP containing IBM graphic codes to ordinary text, allowing printing to printers other than IBM/Proprinter/BJ10E. This program can also be used to strip out Epson codes from other files. |
| CHANGES.D | Details the changes between different versions of PocketD. |
| FEATURE.CMP | This comprehensively compares PocketD's features with other programs such as PCOPY, DX, RIR, etc. This file can be viewed by typing: D /q FEATURE.CMP followed by "L" (list). |
| REGISTER.DOC | This provides details of how to register. |
| VENDOR.DOC | Application for Shareware vendors. |
| INSTALL.BAT | An installation program for PocketD. |
| DEMO.BAT | A demonstration and tutorial for PocketD. |
| *.BAT | Various examples of how PocketD can be used to emulate other utilities such as NCD, WHICH and WHEREIS, etc. (See below.) |

## Additional BAT files

The following BAT files will give usage help if used with the single parameter /?. For example

```
DELBUT /?
```

Most can accept additional options. For example

`FD /T` will select the 1st subdirectory found that was created today.

## The BAT files provided include:

"f1 f2 .." below indicates multiple filespecs as parameters.

| | |
|---|---|
| PsL_UNPK d1 d2 | This will unpack a PsL disk of self-unpacking programs. See D /?? for details. |
| ATTRIB flags f1 f2.. | A full ATTRIB command allowing setting and clearing of all flags. |
| DELBUT f1 f2 .. | Delete all but filespecs f1 f2 .. |
| DELDIR f1 | Recursively delete specified subdirectory. |
| FD f1 f2 .. | Search for and select a subdirectory matching f1 (or f2, etc.). |
| TOUCH f1 f2.. | Touch the matching files to give the current date and time. |
| VDEL f1 f2 .. | Delete all matching files, prompting before each. |
| WHEREIS f1 f2 .. | Search for files/dirs matching filespecs |
| WHICH f1 | Will show from which subdirectory DOS would run the program *f1* (See D /??). |

## 10 Nifty Things You Can Do With PocketD

1. Use it instead of DOS's DIR for viewing your directories.

   Just typing *D* gives you a sorted double column display that pauses when the screen is full. In addition to this, you can add configurable color-coding of filenames and more tailoring and display options than any other directory listing program (as of June '92).

2. Use it to back up your disks.

   ```
   D /WTccu A:
   ```

   would search the whole drive for files modified or created today and copy them to drive A:, providing that either the file does not already exist on A: or is older. (PocketD provides 19 copy sub-options).

3. Look for hard-to-find files.

   ```
   D *ASP* /W!E'Shareware' /c
   ```

   would search the whole drive for files with names containing the string "ASP", where the file is nonexecutable and contains the string "Shareware" inside the file. Any files found can then be optionally viewed and/or copied to the current directory. You can make this search more specific by adding size and date ranges and multiple include/exclude filespecs.

4. Create a BAT file that could rebuild the subdirectory structure of your hard disk.

```
D /WDu[md $w//] > RESTDIR.BAT
```

5. Uncompress each ZIP file on A: into its own subdirectory on the current drive.

```
D a:*.ZIP [MD $n//PKUNZIP $w $n\]R
```

6. Take a PROCOMM log file and rename it to give a unique name based on its date.

```
D PCPLUS.LOG [D $f $d.* [REN $f
   $d.$$F]R.u]R/.m
```

   giving names of the form 92Feb03.1, 92Feb03.2, 92Feb12.1, etc.

7. Search for files in any subdirectory or ZIP, ARC, PAK, LZH, ZOO, or ARJ archive.

```
D *.TXT *=D=D* /WZ
```

   would search the current drive scanning all subdirectories, including any archives found, for any files with the extension .TXT or a consecutive double digit in its name (e.g., DIET14.EXE but not PRO1V2.COM).

8. Analyze your hard disk to find out the relative sizes of each subdirectory

```
D /r%gss
```

   This example also gives the relative percentage usage of each subdirectory and plots a bar chart to show the relative sizes, sorting the output by subdirectory size.

9. To remind you when you haven't backed up your hard disk recently.

```
D BACK.TAG /!:-7 [You have not
   backed-up your disk since $d//]
```

10. Ask multi-option questions from within BAT files (e.g., AUTOEXEC.BAT).

```
D /Ywbd /'Choose (W)indows,
   (B)ackup hard disk or (D)os
   prompt ?'
IF ERRORLEVEL 3 GOTO DOS
IF ERRORLEVEL 2 GOTO BACKUP
IF ERRORLEVEL 1 GOTO WINDOWS
```

## PocketD Registration Form

Name: ______________________________
Company: ___________________________
Address: ___________________________
City: ______________________________
State: __________ Country: ______________
Zip: __________ Phone: ________________

U.K. address for information
and orders in Sterling:
PocketWare
Box 2369, Hendon
London, NW4 1NR, England

U.S. Orders:
PsL
P.O. Box 35705
Houston, TX 77235-5705

You can order in the following ways:

* U.S. dollars with your MC, Visa, AmEx, or Discover card by calling 800-2424-PsL or 713-524-6394 or by FAX to 713-524-6398 or by CIS to 71355,470 or by mail to the PsL address. Quote PsL item #10474. These numbers are for ordering only. For product information, technical support, site licences, etc., write to the U.K. address or contact via CIS 100031,3537.

* Via CompuServe. Type "GO SWREG" and select Reg ID 126(private) and 127(commercial).

* Pounds Sterling by cheque, banker's draft, or money order against any U.K. bank to the U.K. address.

Inquiries are welcomed for bundling with other software (already bundled with PC-ACCESS).

| **PsL item #10474** | Private/Educational | Commercial |
|---|---|---|
| Price per copy | £19.00, $35.00 | £29.00, $54.00 |

Prices include a pocket-sized 110-page manual with each registered copy. See the file REGISTER.DOC for details on multiple-copy orders and site licenses. See the file D.DOC for registration benefits.

Quantity: __________ (or Site) UK or US
Total : ........................................ £ ______ $ ______
Basic S&H : .................................. £ 2.00 $ 4.00
Overseas S&H: .............................. £ ______ $ ______
Total cost: ..................................... £ ______ $ ______

Overseas Shipping: If ordering multiple copies from outside the U.K. (excluding Site Licence purchase)
add for each additional ........... £ 1.70 $ 3.00
Covers cost of shipping multiple manual copies.

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Payment by: ☐ Check ☐ Other
☐ MC ☐ VISA ☐ AmEx ☐ Discover

MC/Visa/AmEx/Discover for US $ to PsL only! (valid for UK Sterling only) Cheq/Draft/MO for UK £ to PocketWare only!

Authorized Signature ____________________
Expiration Date ________________________
Card Number __________________________

# QEdit(R) Advanced

**The "Quick Editor"**
**Version 2.1**
**Copyright © 1985–1991 Applied Systems Technologies, Inc.**
**SemWare and QEdit are registered trademarks of Applied Systems Technologies, Inc.**

## Introduction

"You can never be too rich, too thin, or have too much RAM and Disk storage." Ancient American proverb, ca. 1980.

This manual describes the use and operation of QEdit, a full-screen text editor for the IBM PC and compatible computers. The manual includes installation instructions, a user's guide, and a reference section which explains all of QEdit's commands.

QEdit is a text editor as opposed to a word processor. This means that most of its features are geared toward creating and maintaining program source code and other text files. However, QEdit can also be used like a word processor to write documents or letters since it includes word-wrapping and paragraph reformatting capabilities.

QEdit was developed with these goals in mind:

FAST OPERATION — More than anything else, QEdit was designed to operate quickly, especially in these areas:

- Initially loading a file for editing.
- Displaying/Updating the screen.
- Moving through the editing window.
- Searching for text.
- Saving a file back to disk.

EASE OF USE — This version of QEdit includes:

- Optional *Pop-Down* menus.
- Customizable Help Screen.
- Complete execution from a single file (Q.EXE).
- Pop-up file directory and picker.

COMPACT PROGRAM SIZE — Every effort was made to keep the program size as compact as possible. Even though memory is getting cheaper all the time, we didn't want QEdit to require any more memory than absolutely necessary.

## Features

These are just some of the many features you will enjoy while using QEdit:

- It is very fast!
- It is customizable. QEdit can be easily configured the way you like it. This includes the meaning of ALL special keys on the keyboard (Alt, Ctrl, and function keys; <Home>, <PgUp>, etc.), colors, screen appearance, and many other options.
- It utilizes all available memory. With QEdit you can edit files as large as your computer's memory will allow.
- It allows true multi-file editing. The number of files you can edit is limited only by available memory.
- It enables you to view files through up to eight *windows*. You can have multiple views of the same file or view a different file in each window.
- It provides up to 99 scratch buffers for cut-and-paste or template operations.
- It allows you to exit to DOS (or a DOS shell) temporarily and use DOS commands from within the editor.
- It is easy to use. You can use the optional Pop-Down menus to access QEdit's commands, and there is a Help Screen which you can customize for your own needs.
- It supports Character, Line, and Column Blocks.
- It allows you to create macros which will execute a series of commands and keystrokes by pressing a single key.
- It provides wordwrap and paragraph reformat capabilities.
- It saves deleted words, lines, and Blocks in a deletion buffer for later recall.
- It provides *C mode*, which utilizes automatic indentation for C language programming.

- It locates matching braces and parentheses, a useful feature for programming.
- It will load multiple files from the DOS command line, with or without wildcard characters.
- It allows you to specify up to six default file extensions for use when loading files and determining tab settings.
- It allows you to execute command-line compilers from within QEdit, using its macro capability.
- It provides line drawing to create diagrams and tables.
- It supports enhanced (101 key) keyboards.
- It supports up to 200-column and 100-line screens.
- It supports EGA 43-line mode and VGA 50-line mode.
- It is compact. Even with all these features, QEdit requires less than 50K of disk space.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the QEDIT files in the directory \SECRETS\QEDIT. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the QEDIT files to a directory that is included in the search path.

The following files are included on the distribution diskette.

| *File* | *Description* |
|---|---|
| Q.EXE | The editor program. This is the only file required to run QEdit Advanced. |
| QCONFIG.EXE | The configuration program. This program allows you to customize QEdit. |
| QCONFIG.DAT | The default Keyboard Definition File. |
| QHELP.TXT | The default Help text. |
| QEDIT.DOC | The documentation for the shareware version of QEdit, including LICENSE and WARRANTY information. |
| ORDERFRM.DOC | Handy order form. |
| READ.ME | Read this file first. |
| QEDITTSR.DOC | Release notes about the new TSR version of QEdit. |
| VENDOR.DOC | Restrictions/Authorizations for Disk Vendor distribution. |

## Chapter 1: User's Guide for QEDIT

### Quick start

With QEdit you can edit any text file with a few simple commands. The next three sections briefly describe the quickest way possible to load, edit, and save a file.

#### Getting started

Using QEdit is simple and straightforward.

To initiate QEdit, type **Q <Enter>**. The Editor will respond with a prompt for the name of the file to be edited.

```
File(s) to edit: <Esc = abort>
```

Enter the filename. The filename may include full path designation.

**QEdit Tip:** At any FILE TO EDIT prompt, you may use wildcard characters (* or ?) to obtain a list of matching files in the directory.

You may optionally include the filename on the DOS command line in order to bypass this prompt. For example, from the DOS command line, type

```
Q filename <Enter>
```

#### In the editor

Once you are in QEdit, the cursor movement keys (Left, Right, Up, Down, Home, End, ...) allow you to move around in the text. To obtain a help screen, simply press <F1>. Pressing <F1> again returns you to the text.

To obtain a menu of QEdit commands, press <Esc>. You may then use the cursor movement keys to position the cursor bar over the desired command. Pressing <Enter> will execute the command.

**QEdit Tip:** Pressing <Esc> will abort any editor prompt.

### Getting out

The easiest way to get out of QEdit and save all the work you have done is to enter <Alt F> <G>.

## Menus

QEdit provides an easy-to-use Pop-Down style menu system. The menu system is very helpful while learning to use the editor. It is also handy for executing seldom-used commands.

**To access the menus and execute a command:**

1. Press <Esc>. The main menu will be displayed at the top of the screen. The main menu consists of a list of sub-menu items.
2. Select a sub-menu item. You may do this by moving the cursor bar to the desired item and pressing <Enter> or by typing the capital letter of the desired item.
3. Execute a command. This is done in the same manner as selecting the sub-menu items.

***Note:*** See the file QEDIT.DOC for visual example omitted here.

You can execute many QEdit commands using the menus. As you become more familiar with the editor, you may wish to use the <Ctrl>, <Alt>, and function keys, instead of the menus, to execute commands more quickly. A cross reference of all QEdit commands can be found in Appendix A.

## The status line

At the top of the editing screen is the Status Line. The Status Line constantly displays information regarding the file you are currently editing.

***Note:*** Please see the file QEDIT.DOC for visual example omitted here. It shows the Status Line with a description of the information displayed.

For more information on Insert, AutoIndent, WordWrap, and Box Drawing, refer to the "Modes" section in this chapter. For more information on macro recording, refer to the "Macros" section in this chapter.

**QEdit Tip:** The Status Line may be configured to appear at the bottom of the screen. Refer to the "Colors/Screen Options" section of Chapter 2 for more information.

## Files

### Creating new files

There are three different ways to create new files, ready for editing:

1. From the DOS command line, enter <Q> followed by one or more new filenames. Each filename must be separated by a space. For example, typing

```
Q file1 b:file2 \dir1\file3 <Enter>
```

will create **file1** in the current directory, **file2** on disk drive B, and **file3** in the directory **dir1**.

**QEdit Tip:** When specifying file names to QEdit, you may use full path designations if desired.

2. From the DOS command line, type **Q <Enter>**. QEdit will respond with the prompt:

```
File(s) to edit: <Esc = abort>
```

Enter one or more new filenames, each separated by a space.

3. From within the editor, enter the EditFile <Alt E> command. The above prompt will appear. Again, you may enter one or more new filenames, each separated by a space.

### Loading existing files

Loading existing files works exactly like creating new files (in the previous section) with two exceptions

- The filename(s) specified must exist on disk.
- The filename(s) may contain wildcard characters (*, ?).

QEdit can be configured to respond to wildcarded filename entries in one of two ways:

1. Load all matching files into the editor.
2. Show all the matching filenames in a *pick* listing. You can then use the cursor keys to find the file you want and press <Enter> to load it into the editor.

QEdit can also be configured to use default file extensions, such as DOC and PAS when entering filenames. For more information on configuring these options, refer to the "General Options" section of Chapter 2.

### Multiple files: the ring

QEdit is a true multi-file editor. It allows as many files as will fit into memory to be loaded at the same time. All of these files can then be edited by simply switching back and forth between them without having to save and load each file individually. Switching between files is accomplished very quickly and easily.

A simple explanation of how QEdit handles multiple files will help you in manipulating and editing multiple files.

QEdit maintains all loaded files in a *ring*.

***Note:*** See the file QEDIT.DOC for visual example omitted here.

In the ring we see that six files have been loaded into the editor. You can, of course, load many more files — as many as your computer's memory will allow.

Assume the file currently being edited is file *B*. You may then move from one file to another in any of the following ways:

- Enter the NextFile <Alt N> command. You will now be editing the next file in the ring, file *C*.
- Enter the PrevFile <Ctrl KP> command. You will now be editing the previous file in the ring, file *A*.
- Enter the EditFile <Alt E> command. The editor will prompt for a filename. You may enter the name of any file in the ring. The editor will make the entered file the current one for editing.

You may, at any time, add files to the ring or discard files from the ring. If you add a new file to the ring using the EditFile command, it will become the current file and the old current file will become the previous file. If you delete a file from the ring, the previous file will become the current file.

**QEdit Tip:** When loading files, be aware that QEdit will not load two copies of the same file. If you issue the EditFile command for a file already loaded, that file will become the new current file.

### Saving files and exiting

QEdit provides several different ways to save files, discard files, and exit the editor. A set of five basic commands provides assorted combinations of saving, discarding, and exiting in order to suit individual preferences.

We recommend selecting one or two methods with which you feel most comfortable and using them consistently.

***Note:*** Please see the file QEDIT.DOC for table omitted here. It summarizes these commands and their effects.

## Viewing and getting around in the text

QEdit is quite versatile in allowing you to move around in the text. With simple keystrokes you can either move a few characters or lines at a time or move from one end of the file to another. You can go to specific lines in the file or go to each occurrence of a particular string of characters. And with QEdit's windows, you can view up to eight files at a time or have multiple views of the same file.

As you will see, there are quite a few commands in QEdit for moving around in your text. If you are not already familiar with these types of commands, try them out. You will probably find some new commands that will be useful for your applications of QEdit.

### Cursor movement

#### *Moving through a file*

QEdit provides several commands which enable you to move about in a file.

**BegFile <Ctrl PgUp> and EndFile <Ctrl PgDn>**

As their names imply, these commands move the cursor to the beginning or end of the current file. BegFile positions the cursor at column one on the first line of the file. EndFile positions the cursor after the last nonblank character on the last line of the file.

**PageUp <PgUp> and PageDown <PgDn>**

PageUp scrolls the text toward the top of the file, one page-full, less one line, at a time. PageDown scrolls the text toward the end of the file, one page-full, less one line, at a time.

**HalfPgUp and HalfPgDn**

These commands work exactly like the PageUp and PageDown commands, except that they scroll only one-half page at a time.

**QEdit Tip:** All QEdit commands, including those without default key assignments, can be assigned as the user desires, to almost any key. Refer to the "Keyboard Configuration" section of Chapter 2 for more information.

### ScrollUp <Ctrl W> and LineUp

These commands scroll the text one line at a time toward the beginning of the file. The ScrollUp command will cause the cursor to remain on the same line of text until it reaches the bottom of the screen, while the LineUp command will cause the cursor to remain stationary on the screen.

### ScrollDown <Ctrl Z> and LineDown

These commands scroll the text one line at a time toward the end of the file. The ScrollDown command will cause the cursor to remain on the same line of text until it reaches the top of the screen, while the LineDown command will cause the cursor to remain stationary on the screen.

### GotoLine <Ctrl J>

Upon entering the GotoLine command, QEdit will prompt with:

```
line number: <Esc = abort>
```

By typing a line number and <Enter>, QEdit will *jump* the cursor to the specified line number. An easy way to move about quickly in your text is to keep track of critical positions by noting their line number, which is displayed on the Status Line, and then using this command to jump to those positions.

### GotoBlockBeg <Ctrl QB> and GotoBlockEnd <Ctrl QK>

These commands move the cursor to the beginning or to the end of the currently marked Block. If the marked Block is in another file, QEdit will make that file the current file and then move you to the beginning or end of the Block. For more information on using marked Blocks, refer to the "Block Commands" section in this chapter.

### PrevPosition <Ctrl QP>

This command can come in very handy. It moves you to the cursor's previous position. Although it would not be of much use if you had only moved a character or two, it can be useful for larger jumps of the cursor.

For example, perhaps you are in the middle of a file and want to check something at the very beginning of the file. To get to the beginning, you would simply enter BegFile <Ctrl PageUp>. After you look at the beginning, you want to get back to where you were previously. You could use PageDown, but that could be tedious, especially if your file is large. The easiest way would be to enter PrevPosition, and with a single keystroke, you are immediately back where you started.

***Note:*** PrevPosition can only take you back to the cursor's immediately previous position.

If there are intervening keystrokes that move the cursor, PrevPosition will move you back to the most recent location of the cursor.

### Find <Ctrl QF> and RepeatFind <Ctrl L>

These commands will search for a specific string of characters in the file. Upon entering the Find command, QEdit will prompt for the string of characters for which to search. Search for

```
<Esc = Abort>
```

Enter the desired string of characters and press <Enter>.

QEdit will then prompt for the search options to use.

```
Options [BGLIW] (Back Global Local
   Ignore-case Whole-words):
I
```

Enter the letter(s) corresponding to the option(s) you want and press <Enter>. The available options are (see Appendix E for additional options):

[B] = Search backward from the current cursor position. When you choose *B*, QEdit will search backward from the cursor position toward the beginning of the file. Otherwise, it will search from the cursor position toward the end of the file.

[I] = Ignore the case (capital or lowercase) of the search string. For example, "Hello" would match "hello" if this option is chosen. This option is set ON by default.

[W] = Search for whole words only. For example, when this option is chosen, the word "sent" will match only the actual word "sent". Otherwise, any other words that contain the string (such as "sentence" or "absent") would also match.

Once the Find command is executed, the cursor will be placed at the first occurrence of the search string found in the text. To locate subsequent occurrences of the search string, simply enter the RepeatFind command.

If you want to search for a string and replace it with another string, refer to the "Replacing Text" section in this chapter.

**Match <Alt F3>**

This is a specialized search command. It acts only on the characters ( ), { }, and [ ]. With the cursor placed on one of these characters, enter the Match command. QEdit will locate the logical matching character of the pair. This command can be useful for programming in languages where complicated expressions are grouped using these characters.

### *Moving around on the screen*

These commands affect the cursor position within the current screen of displayed text.

**BegScreen <Ctrl Home> and EndScreen <Ctrl End>**

These commands move you to the first or last line of the screen (or current window if you have windows open). The cursor's column position does not change.

**BegLine <Home> and EndLine <End>**

You are probably already familiar with these commands. BegLine moves the cursor to the beginning, or column one, of the current cursor line. EndLine moves the cursor to the last nonblank character of the current cursor line.

**ScreenLeft <Alt F5> and ScreenRight <Alt F6>**

These commands scroll the entire text to the left or right one column at a time. You can change the number of columns which are shifted each time using the QEdit configuration program. Refer to the "Colors/Screen Options" section of Chapter 2 for more information.

**WordLeft <Ctrl Cursor Left> and WordRight <Ctrl Cursor Right>**

These commands move the cursor to the first character of the previous or following word in the text.

**MakeCtrofScreen <Shift F5>, MakeTopofScreen <F5>, MakeBotofScreen**

These three commands change the position of the cursor line on the screen (or window, if you have windows open). They can be useful if you prefer to work at the bottom or top of the screen or if you want to keep the text surrounding your cursor line in view.

The MakeCtrofScreen command scrolls the text on the screen (either up or down) until the cursor line is the center line of the screen.

The MakeTopofScreen command scrolls the text upward until the cursor line is the top line of the screen.

The MakeBotofScreen command scrolls the text downward until the cursor line is the last line of the screen.

### *Moving by lines and characters*

These commands enable you to move in short increments through the text.

**CursorLeft <Cursor Left> and CursorRight <Cursor Right>**

These commands move the cursor one column to the right or left on the cursor line.

When you use CursorLeft, the cursor will stop when it reaches the left edge of the screen, unless the text has been scrolled to the left. In that case it scrolls the text to the right one column at a time until it reaches column one.

When you use CursorRight and the cursor reaches the right edge of the screen, it will begin to scroll the text to the left. It will stop scrolling when it reaches the maximum line length.

**CursorUp <Cursor Up> and CursorDown <Cursor Down>**

These commands move the cursor up or down in the file one line at a time. The cursor's column position does not change.

## Using windows

QEdit provides the ability to view different areas of files (either multiple files or the same file) at the same time, through the use of windows. A window is a portion of the screen that allows you to view text. You can divide your screen into as many as eight horizontal windows. You can then view as many as eight different files through these windows or have multiple views of a single file. This capability can be quite helpful for comparing text, copying text, and moving text.

***Note:*** Please see the file QEDIT.DOC for split-window example omitted here.

### *Creating windows*

To create a window, use the HorizontalWindow <Ctrl OH> command. The screen will be split into two halves, or windows, each window having its own Status Line. The cursor will reside in the newly created window, making it the current window.

If only one file is being edited, the newly created window will simply be an additional view of the same file. QEdit will not load more than one copy of the same file.

If multiple files are being edited, the newly created window will be a view of the next file in the ring. (Refer to the “Multiple Files” section in this chapter.) If desired, you can configure QEdit to prompt for the name of the file to be viewed in the new window instead of the editor automatically selecting the next file in the ring. Refer to the “General Options” section of Chapter 2 for more information.

Once the screen has been split into windows, entering the HorizontalWindow command again will affect the current window. All other windows will remain unchanged.

### *Switching windows*

When multiple windows are opened, editing will only take place in the current window (the window containing the cursor). For editing text in other windows, two commands are provided to switch from one window to another.

- The PrevWindow <Ctrl OP> command will move the cursor to the window above the current window. If the current window is at the top of the screen, the cursor will move to the last window on the screen.
- The NextWindow <Ctrl ON> command will move the cursor to the window below the current window. If the current window is at the bottom of the screen, the cursor will move to the first window on the screen.

### *Changing window size*

You can change the size of any of the windows on the screen with the commands below.

- The GrowWindow <Ctrl OG>, ShrinkWindow <Ctrl OS>, and ResizeWindow <Esc><W><R> commands allow you to change the size of the current window. Upon entering any of these commands, QEdit will prompt you to change the size of the window by using the cursor up and cursor down keys.
- The ZoomWindow <Ctrl OZ> command will cause the current window to fill the entire screen as all other windows disappear. To restore the screen with all windows, simply press <Ctrl OZ> again. Be careful not to confuse this command with the OneWindow command (see below). OneWindow fills the screen with the current window (like ZoomWindow does), but it also closes all windows except the current one. ZoomWindow leaves the other windows intact, even though they are hidden from view.

### *Closing windows*

There are two commands to close windows.

- The CloseWindow <Ctrl OC> command closes the current window. The current window will disappear and the cursor will be placed in the window above the closed window, making it the new current window. When the closed window is at the top of the screen, the window below the closed window will become the new current window.
- The OneWindow <Ctrl OO> command closes all windows except the current window. The current window will fill the entire screen. Note that closing a window does not discard files from the editor.

## Copying, moving, replacing, and deleting text

### Block commands

In this section, we will see how to mark and manipulate Blocks. A Block is some portion of the text file which has been specifically delineated using QEdit's Block marking commands. We will see how to manipulate Blocks both directly and through the use of intermediate storage areas known as buffers.

QEdit contains a rich set of Block commands. There are commands to mark Blocks by lines, columns, and characters. There are commands to copy, move, shift, and delete Blocks. These commands are very useful for transferring text from one file to another. There are also cut-and-paste commands, as well as commands to move Blocks to and from named Scratch Buffers.

#### *Marking and unmarking a block*

Before you can manipulate a Block, you must first *mark* or outline a Block. There are four ways of marking a Block in QEdit. The one you choose will depend on the editing situation and what you find most comfortable to use.

**MarkLine <Alt L>**

This command allows you to mark a Block in whole line increments. In other words, the Block will contain complete lines only and no portions of lines. To use this command, place the cursor anywhere on the first line of the text you wish to mark and press <Alt L>. (Notice that the line is immediately marked.) Now, move the cursor (the Block will *follow* the cursor) to the last line of text to be marked and press <Alt L> again. You now have a marked Block, ready for manipulation.

**DropAnchor <Alt A>**

This command allows you to mark a Block one character at a time. In other words, the Block will contain a stream of characters (which can span over multiple lines). To use this command, place the cursor over the first character of the text you wish to mark and press <Alt A>. Then begin moving the cursor toward the end of the text to be marked. Notice that the Block follows the cursor position. Once the cursor is placed over the last character to be marked, press <Alt A> again. You now have a marked Block, ready for manipulation.

**MarkCharacter**

This command works just like DropAnchor except that while marking the Block, the cursor is not included within the Block.

MarkCharacter is provided as a replacement for the DropAnchor command. If you want the cursor position included in character Blocks, then continue to use DropAnchor; otherwise, replace the DropAnchor command with this command in the QCONFIG.DAT file. Refer to the "Keyboard Configuration" section of Chapter 2 for more information on changing QEdit command assignments.

**MarkColumn <Alt K>**

This command allows you to mark one or more columns of text. To use this command, place the cursor over the upper left-hand character of the text you wish to mark and press <Alt K>. Then begin moving the cursor to the right and/or downward. Notice that the Block follows the cursor position. Once the cursor is placed over the lower right-hand character of the Block, press <Alt K> again. You now have a marked Block, ready for manipulation.

**QEdit Tip:** When using the MarkLine, MarkCharacter, MarkColumn, and DropAnchor commands, it is not necessary to mark the end of the Block. QEdit will assume the end of the Block to be the current cursor position for DropAnchor, MarkCharacter, and MarkColumn and the current cursor line for MarkLine. Marking the end of the Block is only necessary when moving or copying text within the same file.

**MarkBlockBegin <Ctrl KB> and MarkBlockEnd <Ctrl KK>**

This is another method of marking a Block one character at a time. To mark a Block with this command, place the cursor over the first character of the text you wish to mark and press <Ctrl KB>. Then position the cursor one character past the end of the text that is to be marked and press <Ctrl KK>. The Block is now marked and ready for manipulation.

**UnmarkBlock <Alt U>**
Entering the UnmarkBlock command causes the currently marked Block to be unmarked.

QEdit allows you to mark one Block at a time. If you mark a Block and then go elsewhere in the file and attempt to mark another Block, QEdit will simply extend the first Block by including all the text between the original Block and the current cursor position. If you mark a Block in one file and then go to another file and mark a Block, QEdit will automatically unmark the Block in the first file.

Also, if you mix types of Block marking, by entering MarkLine followed by DropAnchor or vice-versa, QEdit will mark the Block according to the last command entered.

### *Manipulating blocks directly*

QEdit provides several ways to manipulate a marked Block. The following commands act directly on a marked Block.

**QEdit Tip:** When Copying or Moving Blocks, blocks marked using the DropAnchor, MarkCharacter, MarkColumn, and MarkBlockBegin/End commands will always be inserted beginning at the current cursor position. Blocks marked using the MarkLine command will be inserted at either the line before or the line after the current cursor line. Refer to the “General Options” section of Chapter 2 for more information.

**CopyBlock <Alt C>**
This command makes a copy of the marked Block and inserts it where you decide. You can insert this copy of the Block at some other place in the same file or in another file. The original Block of text, from which you made the copy, is not affected.

To use this command, first mark the Block as described above. Next, move the cursor to the position in the file where you wish to insert the marked text. (Or, if you have multiple files loaded and wish to copy text from one file to another, use the NextFile, PrevFile, or EditFile commands to get a new current file and likewise position the cursor). Now enter the CopyBlock <Alt C> command. The Block will be inserted at the new position. You may notice, after you use CopyBlock, that the newly inserted Block of text is now marked and the original Block is unmarked. To unmark the copied Block, enter the UnmarkBlock command.

QEdit provides an easy method for copying the same Block repeatedly. Refer to the Copy and Paste commands in the following section for more information.

**CopyOverBlock <Alt Z>**
This command is for use with column Blocks only. It works just like CopyBlock except the Block is copied to the current cursor position by overlaying existing text and without shifting text to the right.

**MoveBlock <Alt M>**
This command works exactly like CopyBlock with one exception: Upon entering the MoveBlock <Alt M> command, the originally marked Block is deleted from the file.

As with the CopyBlock command, you can move a Block to another part of the same file or from one file to another.

QEdit provides an easy method for deleting a Block from its original position and inserting it repeatedly at other positions.

Refer to the Cut and Paste commands in the following section for more information.

**DeleteBlock <Alt G>.**
This command simply deletes a marked Block of text from the file. To use this command, first mark a Block of text to be deleted, then enter the DeleteBlock command. The marked text will be deleted from the file. (Deleted Blocks will be placed in the Kill buffer. Refer to “The Kill Buffer” section in this chapter for more information.)

**ShiftLeft <Shift F7> and ShiftRight <Shift F8>**
These commands allow the User to shift text contained in a Block to the left or right. To use these commands, first mark a Block. Now enter ShiftLeft or ShiftRight. The entire marked Block of text will be shifted one column to the left or right respectively. If there is no marked Block of text or the cursor is outside of the marked Block, the current cursor line will be shifted. This command is very useful for changing indentation for portions of text or source code.

### *Manipulating blocks using scrap buffers*

QEdit provides an extremely helpful device for quickly manipulating Blocks of text. This device consists of a Scrap Buffer and four QEdit commands which act on the Scrap Buffer.

The Scrap Buffer is a temporary holding area for marked Blocks of text. The commands Cut, Copy, Paste, and PasteOver are solely responsible for placing text into the Scrap Buffer and copying text from the Scrap Buffer.

When you issue a Copy or Cut command (we will discuss how to use these in the following section), the marked Block is placed into the Scrap Buffer. The next time you Copy or Cut a Block into the Scrap Buffer, the previous contents of the Scrap Buffer are deleted and replaced with the new Block.

You can repeatedly use the Paste and PasteOver commands as many times as needed to insert a copy of the Block held in the Scrap Buffer at multiple positions in your file or files. The Paste and PasteOver commands do not purge the contents of the Scrap Buffer.

#### Copy <grey +> and Paste <grey *> or PasteOver <Ctrl PrtSc>

To copy a section of text from its present position and insert it somewhere else:

1. Mark the text using the Block marking commands.
2. Enter Copy <grey +>. This places a copy of the marked text into the Scrap Buffer.
3. If you want to insert the text somewhere else within the same file, use the cursor movement commands to place the cursor where you wish to insert the text.
4. If you want to insert the text in another file, switch to that file using the NextFile, PrevFile, or EditFile command to access that file. Then, likewise place the cursor at the desired position.
5. Enter Paste <grey *> to insert the text. If you have copied a column Block into the Scrap Buffer and wish to insert it by overlaying existing text then use PasteOver <Ctrl PrtSc> instead of Paste.

Since Paste and PasteOver leave the contents of the Scrap Buffer intact, you can make multiple insertions in the same or different files. Simply move the cursor to the file and position where you wish to make additional insertions and enter Paste <grey *> or PasteOver <Ctrl PrtSc>. Please note that the next time you issue a Cut or Copy command, the previous contents of the Scrap Buffer will be discarded.

#### Cut <grey -> and Paste <grey *> or PasteOver <Ctrl PrtSc>

Cut and Paste/PasteOver work exactly like Copy and Paste/PasteOver except that Cut will delete the marked Block from the file. You can still make multiple insertions of the text using the Paste or PasteOver Commands.

**QEdit Tip:** QEdit can be configured to have the Cut and Copy commands act on the current cursor line if no Block is marked. Refer to the "Advanced Options" section of Chapter 2 for more information.

### *Manipulating blocks using scratch buffers*

A Scratch Buffer is a special type of buffer to which you assign a name. QEdit allows you to create and name up to 99 Scratch Buffers for each editing session. These can be useful if you have several different Blocks of text that you want to insert in multiple locations. The commands StoreScrbuff <Ctrl BS>, AppendScrbuff<Ctrl BA>, and GetScrbuff <Ctrl BL> are used to place text in, and retrieve text from, a Scratch Buffer. Whenever you issue one of these commands, QEdit will ask you for the name of the Scratch Buffer. The contents of all Scratch Buffers are discarded when the editor is terminated.

#### StoreScrbuff <Ctrl BS> and AppendScrbuff <Ctrl BA>

To place a Block of text into a named buffer:

1. Mark the Block.
2. If you want to create a new Scratch Buffer containing the marked Block, enter StoreScrbuff <Ctrl BS>. (This command is also used if you wish to replace the current contents of a Scratch Buffer with the marked Block). If you want to append the marked Block to the current contents of the Scratch Buffer, enter AppendScrbuff <Ctrl BA>.

3. QEdit will prompt for the name of the Scratch Buffer.

```
Store (Append) to buffer: <Esc =
   abort>
```

4. Enter a name. It is best to use a name which reflects the contents of the buffer and is easily remembered.

The Block is now stored in a Scratch Buffer under the assigned name.

**GetScrbuff <Ctrl BL>**

To retrieve the contents of a named Scratch Buffer and insert it at the current cursor position:

1. Position the cursor where you wish the contents of the named Scratch Buffer to be inserted.
2. Enter GetScrbuff <Ctrl BL>. QEdit will prompt for the name of the Scratch Buffer to be retrieved.

```
Get from buffer: <Esc = abort>
```

3. Enter the name of the desired Scratch Buffer.
4. Repeat the above steps whenever you want to insert the contents of a Scratch Buffer into your text.

### *Examples of block commands*

This section contains examples of commonly used Block Commands.

**Copying or moving a Block within the same file.**

1. Position the cursor at the beginning of the Block to be copied or moved.
2. Enter DropAnchor <Alt A> to mark a stream of characters, or MarkLine <Alt L> to mark whole lines, or MarkColumn <Alt K> to mark a column of text.
3. Position the cursor at the end of the Block to be copied or moved.
4. Enter DropAnchor, MarkLine, or MarkColumn again to mark the end of the Block.
5. Move the cursor to the location in the file where you want to insert the marked Block.
6. Enter CopyBlock <Alt C>, CopyOverBlock <Alt Z> (column Blocks only), or MoveBlock <Alt M>.
7. Enter UnmarkBlock <Alt U> to unmark the Block.

The Block has now been copied or moved.

**Copying or moving a Block to another file.**

1. Position the cursor at the beginning of the Block to be copied or moved.
2. Enter DropAnchor <Alt A> to mark a stream of characters, or MarkLine <Alt L> to mark whole lines, or MarkColumn <Alt K> to mark a column of text.
3. Position the cursor at the end of the Block to be copied or moved.
4. Switch to the *target* file using the file commands EditFile, NextFile, or PrevFile.
5. Move the cursor to the location in the file where you want to insert the marked Block.
6. Enter CopyBlock <Alt C>, CopyOverBlock <Alt Z> (column Blocks only), or MoveBlock <Alt M>.
7. Enter UnmarkBlock <Alt U> to unmark the Block.

The Block has now been copied or moved.

**Copying or moving a Block for multiple insertions in the same file.**

If you want to insert the same Block at several positions within a file, follow these steps:

1. Position the cursor at the beginning of the Block to be copied or moved.
2. Enter DropAnchor <Alt A> to mark a stream of characters, or MarkLine <Alt L> to mark whole lines, or MarkColumn <Alt K> to mark a column of text.
3. Position the cursor at the end of the Block to be copied or moved.
4. If you want to leave the Block in its original position and insert copies of it at other locations, then enter Copy <grey +>.

If you want to delete the Block from its original position and insert copies of it at other locations, then enter Cut <grey ->.

The Block is now stored in the Scrap Buffer.

5. Move the cursor to the location in the file where you want to insert the Block.
6. Enter Paste <grey *> or PasteOver <Ctrl PrtSc> (column Blocks only).

The Block is now inserted. Repeat steps 5 and 6 for each additional insertion.

**Copying or moving a Block for multiple insertions in more than one file.**

If you want to insert the same Block at several positions in more than one file, follow these steps:

1. Position the cursor at the beginning of the Block to be copied or moved.
2. Enter DropAnchor <Alt A> to mark a stream of characters, or MarkLine <Alt L> to mark whole lines, or MarkColumn <Alt K> to mark a column of text.
3. Position the cursor at the end of the Block to be copied or moved.
4. If you want to leave the Block in its original position and insert copies of it at other locations, then enter Copy <grey +>.

If you want to delete the Block from its original position and insert copies of it at other locations, then enter Cut <grey ->.

The Block is now stored in the Scrap Buffer.

5. Switch to the target file using the file commands EditFile, NextFile, or PrevFile.
6. Move the cursor to the location in the file where you want to insert the Block.
7. Enter Paste <grey *> or PasteOver <Ctrl PrtSc> (column Blocks only).

The Block is now inserted. Repeat steps 5-7 for each additional insertion.

**Copying multiple Blocks for insertion in one or more files.**

If you have several Blocks that you want to insert in several different files, follow these steps:

1. Position the cursor at the beginning of the first Block.
2. Enter DropAnchor <Alt A> to mark a stream of characters, or MarkLine <Alt L> to mark whole lines, or MarkColumn <Alt K> to mark a column of text.
3. Position the cursor at the end of the first Block.
4. Enter AppendScrbuff <Ctrl BA> if you want to append to an existing Scratch Buffer. Otherwise, enter StoreScrbuff <Ctrl BS> to create or overwrite an existing Scratch Buffer.
5. QEdit will prompt for the name of the buffer. Enter a name. (It is best to use a name which reflects the contents of the buffer and is easily remembered).

The Block is now stored in a Scratch Buffer under the assigned name. Repeat steps 1-5 for each additional Block you want to store.

6. Switch to the target file using the file commands EditFile, NextFile, or PrevFile.
7. Move the cursor to the location in the file where you want to insert a Block.
8. Enter GetScrbuff <Ctrl BL>. QEdit will prompt for the name of the Scratch Buffer. Enter the name of the desired Scratch Buffer for insertion.

Repeat steps 6 through 8 for each insertion of a stored Block.

***Note:*** Please see QEDIT.DOC for more information on the following topics omitted here: non-block commands, copying text, moving text, deleting and undeleting text, kill buffers, macros, printing, modes, etc.

One interesting command to make note of is ToggleBoxDraw:

**ToggleBoxDraw <Shift F1>**

This is a specialized command which is really a mode of operation that can be switched ON or OFF. If switched ON, the cursor movement keys (up, down, right, left) can be used to *draw* lines in the text. This can be very useful for creating graphs and tables. To switch Box Drawing mode ON, enter <Shift F1>. To switch OFF, enter <Shift F1> again.

## Word processing commands

QEdit has all the features you need to do basic word processing. In addition to the commands which enable you to view and edit your text (discussed in previous sections), QEdit provides word-wrapping and paragraph reformatting capabilities. Also, Appendix E contains information on QEdit commands for changing case, centering a line (or block of lines), and filling a block.

**The right margin**

To benefit from the word-wrap and paragraph reformatting features of QEdit, an appropriate right margin must first be set. The default setting is column 60. This may be changed using the configuration program. (Refer to the "General Options" section of Chapter 2 for instructions on how to change the right margin setting.) This value of the right margin will be used each time the editor is initiated.

QEdit provides the ability to change the right margin setting temporarily while the editor is running without affecting the permanent setting. To change the right margin setting temporarily, enter the SetRmargin <Ctrl OR> command. QEdit will prompt with:

```
right margin [1..300]:
   <Esc = abort>
60
```

Enter the desired column number for the right margin. The new setting will be in effect until the SetRmargin command is entered again or the next time the editor is initiated. WordWrap mode is automatically switched ON whenever the right margin is set using this command.

**WordWrap**

WordWrap is a QEdit mode which can be switched ON and OFF using the ToggleWordwrap <Ctrl OW> command. When switched ON, this mode will cause the cursor to advance automatically to the next line when text is being entered.

The cursor will advance to the next line based on the right margin, which is set using the SetRmargin <Ctrl OR> command. Once the cursor reaches the right margin and a nonblank character is typed, the cursor, along with the word (i.e., any continuous string of nonblank characters) currently being typed, will be advanced to the next line.

The cursor will not advance to the next line if the current line does not contain at least one space.

**Paragraph Reformatting**

The WrapPara <Alt B> command will reformat text from the current cursor position to the next blank line or the end of the file.

The right-hand side of the text will be reformatted based on the right margin. The right margin is set using the SetRmargin <Ctrl OR> command.

The left-hand side of the text will be reformatted based on AutoIndent mode. If AutoIndent mode is switched OFF, the left margin will be column one. If AutoIndent mode is switched ON, the first nonblank character on the line immediately AFTER the cursor line will be used for the left margin. Using the line AFTER the cursor line allows for special indentation for the first line of a paragraph.

***Note:*** See the file QEDIT.DOC for Chapters 2 and 3 omitted here. Chapter 3 includes a more detailed command reference than the one from Appendix A included here.

# APPENDIX A. List Of Commands

## By function

Menu (Esc) Default Customized
Command Sequence Keystroke Keystroke

*Cursor Movement*

| | |
|---|---|
| BegFile | Ctrl PgUp |
| BegLine | Home |
| BegScreen | Ctrl Home |
| CursorDown | Cursor Down |
| CursorLeft | Cursor Left |
| CursorRight | Cursor Right |
| CursorUp | Cursor Up |
| EndFile | Ctrl PgDn |
| EndLine | End |
| EndScreen | Ctrl End |
| GotoBlockBeg | Ctrl QB |
| GotoBlockEnd | Ctrl QK |
| GotoLine | Ctrl J |
| HalfPgDn | |
| HalfPgUp | |
| LineDown | |
| LineUp | |
| MakeBotofScreen | |
| MakeCtrofScreen | Shift F5 |
| MakeTopofScreen | F5 |

| | |
|---|---|
| PageDown | PgDn |
| PageUp | PgUp |
| PrevPosition | Ctrl QP |
| ScreenLeft | Alt F5 |
| ScreenRight | Alt F6 |
| ScrollDown | Ctrl Z |
| ScrollUp | Ctrl W |
| WordLeft Ctrl | Cursor Left |
| WordRight Ctrl | Cursor Right |

*Editing*

| | | |
|---|---|---|
| AddLine | EA | F2 |
| Align | | |
| Backspace | | Backspace |
| DelCh | | Del |
| DelLine | | ED Alt D |
| DelLtWord | | Ctrl Backspace |
| DelRtWord | | Ctrl T |
| DelToEol | EE | F6 |
| DupLine | F4 | |
| GetPrev | Ctrl — (dash) | |
| InsertLine | EI | Alt F2 |
| JoinLine | EJ | Alt J |
| Literal | | Ctrl P |
| Return | | Enter |
| SplitLine | ES | Alt S |
| UnKill | EU | Ctrl U |

*Block/Scratch Buffer*

| | | |
|---|---|---|
| AppendScrbuff | | CtrlBA |
| Copy | | grey + |
| CopyBlock | BC | Alt C |
| CopyOverBlock | | Alt Z |
| Cut | | grey - |
| DeleteBlock | BD | Alt G |
| DropAnchor | | Alt A |
| GetScrbuff | | Ctrl BL |
| MarkBlockBegin | BB | Ctrl KB or F7 |
| MarkBlockEnd | BE | Ctrl KK or F8 |
| MarkCharacter | | |
| MarkColumn | | Alt K |
| MarkLine | | BL Alt L |
| MoveBlock | | BM Alt M |
| Paste | | grey * |
| PasteOver | | Ctrl PrtSc |
| ShiftLeft | | Shift F7 |
| ShiftRight | | Shift F8 |
| StoreScrbuff | | Ctrl BS |
| UnmarkBlock | BU | Alt U |

*File*

| | | |
|---|---|---|
| ChangeFilename | FC | Alt O or Alt F,C |
| EditFile | FL | Alt E or Alt F,L |
| Exit | | Ctrl KD |
| File | FF | Ctrl KX or Alt F,F |
| GExit | QX | Alt X or Alt Q,X |
| GFile | F | Alt F,G |
| GPQuit | QQ | F3 or Alt Q,Q |
| GSave | | Alt Y |
| KillFile | | Ctrl KZ |
| NextFile | FN | Alt N or Alt F,N |
| PQuit | FQ | Ctrl KQ or Alt F,Q |
| PrevFile FP Ctrl KP or Alt F,P | | |
| Quit | | |
| ReadBlock | FR | Alt R or Alt F,R |
| SaveFile | FS | Ctrl KS or Alt F,S |
| WriteBlock | FW | Alt W or Alt F,W |

*Window*

| | | |
|---|---|---|
| CloseWindow | WC | Ctrl OC |
| GrowWindow | WG | Ctrl OG |
| HorizontalWindow | WS | Ctrl OH |
| NextWindow | WN | Ctrl ON |
| OneWindow | WO | Ctrl OO |
| PrevWindow | WP | Ctrl OP |
| ShrinkWindow | WR | Ctrl OS |
| ZoomWindow | WZ | Ctrl OZ |

*Macro*

| | | |
|---|---|---|
| CurrentFilename | | |
| ExecuteScrap | | Ctrl Enter |
| MacroRead | MR | |
| MacroRecord | MM | Ctrl M |
| MacroWrite | MW | |

*Printing*

| | | |
|---|---|---|
| PrintAll | PA | Alt P,A |
| PrintBlock | PB | Alt P,B |
| PrintEject | PF | Alt P,F |
| SetPrintLeftMargin | PL | Alt P,L |
| SetPrintPageSize | PP | Alt P,P |

*Tab*

| | | |
|---|---|---|
| SetCtabwidth | OC | |
| SetPtabwidth | OP | |
| TabLt Shift | | Tab |
| TabRt | | Tab |

*Find/Replace*

| | | |
|---|---|---|
| Find | SF | Ctrl QF |
| FindReplace | SR | Ctrl QA |
| RepeatFind | SA | Ctrl L |

*Toggles*

| | | |
|---|---|---|
| ToggleBackups | OB | |
| ToggleBoxDraw | | Shift F1 |
| ToggleEnterMatching | OE | |
| ToggleIndent | OA | Ctrl QI |
| ToggleInsert | OI | Ins |
| ToggleSmartTabs | | Ctrl QT |
| ToggleTabsExpand | | Alt V |
| ToggleTabsOut | | AltI |
| ToggleWordwrap | OW | Ctrl OW |

*Other*

| | | |
|---|---|---|
| Dos | | Alt F9 |
| Escape | | Escape |
| InsertDate | | |
| InsertTime | | |
| MainMenu | | Escape |
| Match | | Alt F3 |
| QuickHelp | | Alt H or F1 |
| SetEGA25 | | |
| SetEGA43 | | |
| SetRmargin | OR | Ctrl OR |
| Shell | FO | F9 or Alt F,O |
| ShowEntryScreen | | Alt F10 |
| ToggleEGA43 | | Ctrl F1 |
| UndoCursorline | | Ctrl QL |
| WrapPara | | Alt B |

See the file QEDIT.DOC for an alphabetical listing of commands and for Appendices B–E omitted here.

## QEDIT Advanced 2.15 Registration Form

Name: ____________________
Company: ____________________
Address: ____________________
City: ____________________
State: ________ Country: ____________
Zip: ________ Phone: ____________

**Remit to:**

SemWare Order Form #9108-OF04
4343 Shallowford Rd, Suite C3A
Marietta, GA 30062-5022 USA

We take Visa, Master Card, American Express, Diners Club, Carte Blanche, and Discover. You can use this handy order form to order by MAIL, or by:

PHONE: (404) 641-9002 (Mon-Fri, 9 a.m.-5 p.m. ET)
FAX: (404) 640-6213
BBS: (404) 641-8968 (1200/2400/9600 HST, <N81>)
COMPUSERVE: 75300,2710

NOTE: We offer quantity discounts and multiple-user licenses. We also offer student/educational discounts, discounts to BBS sysops, US and Georgia govt. agencies, and approved User Groups. Call for details.

SemWare, send me:

QEdit Advanced with Printed Manual — includes single-user licensed copy of software (Version 2.15) with 90-day limited warranty, spiral-bound indexed manual, reference card, low-cost upgrades.
each ........................................ $54.95 $ ______

SHIPPING, each copy
US/Canada/Mexico ........................ $3.00 $ ______
Overseas ........................................ $10.00 $ ______

Purchase Orders under $100.00, add Billing Fee. (DOES NOT APPLY to PREPAID Check or Charge Card orders.)
Billing Fee ........................................ $6.00 $ ______

Georgia Residents please add Sales Tax
GA Sales Tax ........................................ 5% $ ______
TOTAL (U.S. funds drawn on U.S. bank) ... $ ______

Payment by:
☐ Check ☐ Visa ☐ MC ☐ AmEx ☐ DC
☐ CB ☐ Discover ☐ Purchase order # ______
Authorized Signature: ____________________
Expiration Date: ____________________
Card Number: ____________________

PRICES SUBJECT TO CHANGE WITHOUT NOTICE. Call to confirm current prices. TSR and OS/2 VERSIONS AVAILABLE!

# QFIND

Version 2.22

Copyright © 1992 by Online Reference Software

## Introduction

QFIND is an ultra high-speed file finding program. It searches the disk drive(s) for the specified files using the standard DOS drive, filename and wildcard structure.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the QFIND files in the directory \SECRETS\QFIND. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the QFIND files to a directory that is included in the search path.

The registered version will optionally search ZIP files. The unregistered version contains the following files:

| *Filename* | *Contents* |
|---|---|
| README | Shareware distribution instructions |
| README.PRN | Document printing instructions |
| QFIND.DOC | This file |
| QFIND.EXE | Uses intelligent output display |
| QFIND.HST | History file QFIND versions |
| QFIND.HDR | CompuServe file description header |
| QFIND.KEY | CompuServe file description keyboards |
| QINST.EXE | Installation and customization program |
| SNAPSHOT.EXE | Capture disk sectors for bug reports |
| ORDER.FRM | Ordering and registration form |

## Enhancements

QFIND has the following enhancements:

- ZIP file search
- Fuzzy name search
- Installation program
- Speed
- Color
- Single or multiple drive search
- Elapsed time
- Exclusive hidden file search
- Standard DOS display format
- Intelligent output display
- Support 80x43 and 132x43 screen sizes
- EMS memory used when available
- DOS 5 extended disk partition support
- Dynamic data segment allocation
- Debugging aid
- DesqView aware

## Features

Standard features include the following:

- Install program
- Speed
- Automatic search of multiple drives
- Automatic wildcard names
- Colors
- Elapsed Time
- Display uses DOS directory format
- Intelligent I/O
- Long screen support
- Dynamic memory allocation
- EMS memory support
- Extended disk partitions
- SNAPSHOT debugging aid
- DesqView aware

Registered features include the following:

- Fuzzy search for embedded name characters
- Search single drive only
- Hidden-only file search
- Pause when screen is full
- Verbose path display
- Goto first match
- Scroll buffer
- CGA snow check control
- ZIP file search

## Colors

The user may configure the text and inverse video colors used by QFIND during operation. The banner colors are fixed and cannot be changed. The default value is to accept whatever colors are found at the cursor location when QFIND loads. Invoke QINST and choose the Customize Colors option from the main menu. The Normal Text and Inverse Video data are displayed in the colors currently installed in QFIND. The uninstalled version of QFIND defaults to whatever colors are present at the cursor when the program is invoked. QINST provides an option on the Colors Menu to restore the defaults. QINST displays the default colors in black and white until they are changed. The demo strings are displayed in the new color after each selection. The colors are not installed in QFIND until the user exits QINST. The user is prompted to enter the hexadecimal values for the text or inverse video option. The number is always entered as a 2-digit number. Results are unpredictable if a single digit is entered. The left number is the Background color. The right number is the Foreground color.

## Following are the color option settings:

07 = White text on black
17 = White text on blue
70 = Black text on white (inverse)

The user may enter any value up to FF if the blink is desired. The color banners are fixed colors and may not be changed by the user. TEXT colors are for the filename matches displayed when QFIND executes. A file that matches the input argument will be displayed as a TEXT color even if it is a ZIP file. INVERSE colors are for the subdirectory names and/or ZIP filenames that are displayed. A ZIP file holding a member file that matches the input argument is displayed in INVERSE color. The member will be displayed in TEXT color.

## Drive range

The user may configure QFIND to stop searching after a specified drive letter. This will be useful to those who have a network drive and don't wish to search all logical drives. Floppy drives that have been ASSIGNED to a higher letter may also be bypassed using this configuration option. The default is for drive 'Z.' Invoke QINST and choose the Drive Search Range option from the main menu. The current highest drive will display as a drive letter. The default value in QFIND is 'Z.' The default value for the starting drive is the first hard drive. QFIND begins its search with the first hard drive unless the command-line filename contains a drive letter or the first starting drive is set with QINST. In either case, the search will automatically increment the drive number until no more drives are left to search. Any drive that appears as a valid DOS disk will be searched. If the user has ASSIGNED a floppy drive as a different drive with a letter higher than the last hard disk, it too will be searched. QFIND will pause for a moment if the floppy is empty and then it will quit. Installing the highest search drive option will stop QFIND from searching assigned drives. Another use is on a very large drive that goes from C to W or even higher. In spite of QFIND's tremendous speed, it will still slow down on a very large drive, particularly so, if that drive contains many ZIP files. The user can limit the search range by using the Highest Search option.

## Switches

The user may configure QFIND for any of the command-line switches. These switches are actually toggles, and may be overridden on the command line. If the /Z switch is set by QINST, using it on the command line will toggle it to OFF for a one-time run of QFIND.

## CGA snow checking

QINST is invoked without a command-line name. An optional command line argument [/OFF] will bypass CGA snow checking for machines that have problems with it or don't need it.

**Example:** QINST /OFF

This switch will force QINST to run with snow checking disabled. The user may install this option into QFIND by selecting the SWITCHES entry on the first menu.

## Performance

QFIND is significantly faster than other file finders.

The following times were derived from the OLR BBS server. Files=32, Buffers=25, No cache, Defragged disk.

827 - normal files
712 - zip files
7259 - zip members

16 MHz 80286
82MB C:
DOS 5.0

Zip Search

| | | |
|---|---|---|
| **QFIND222** | **:39** | **1107%** |
| **PKFIND10** | **7:12** | **100%** |
| **SST 53a** | **7:12** | **100%** |

Normal Search

| | | |
|---|---|---|
| **QFIND222** | **1.65** | **276%** |
| **PKFIND10** | **4.56** | **100%** |
| **SST 53a** | **5.47** | **83%** |

Other file finders were tested but not included. They were unable to read the ZIP files or crashed under the large disk partition.

### DOS automatic wildcards

QFIND uses the DOS wildcard system for command-line arguments.

FILENAME. = FILENAME.
FILENAME = FILENAME.???
FILE = FILE????.???
FILE. = FILE .???
.EXE = ????????.EXE
*123.*45 = ????????.???
. = ????????.
.* = ????????.???
*.* = ????????.???
* = ????????.???
*. = ????????.
*.X* = ????????.X??

## Elapsed time

QFIND displays the total amount of time required to search all drive(s).

Totals are displayed for number of files matching the input argument, total number of files, and number of directories.

## DOS directory format

QFIND displays all file data in the DOS directory format. All file data exactly matches the DOS location and format conventions.

The file attributes are appended to the standard DOS display as both a hex number and their associated name(s).

## Intelligent output display

QFIND determines if DOS I/O redirection to a disk file or printer is active.

When redirection is active, all output is displayed a second time using DOS. All screen output except the author banners are then output to the redirected device or file. For the screen, the color attribute found at the starting cursor location is used.

QFIND automatically uses whichever video page number is active at start-up. The user video configuration remains unchanged.

## Long screen support

QFIND uses the screen mode that is active at start-up. The 80x43 and 132x43 text screens are supported. 40 column mode is not supported.

## Dynamic memory allocation

Internal data segment sizes are allocated on the fly for each logical drive that is searched. Memory is allocated in the exact amount required for each drive.

Assume no EMS memory, extended disk partitions and options enabled for a worst-case memory map:

13K code and stack space
128K FAT table
64K path table
64K scroll buffer
64K minimum directory data
16K floating zip segment

349K memory requirements

## EMS memory support

QFIND will utilize up to 12 pages of EMS memory for the path table and scroll buffer. This increases the directory data space by 132K, and reduces a potential out-of-memory error when searching large subdirectories.

QFIND identifies the EMMXXXX0 device drive signature and then allocates EMS memory via DOS interrupt 67h with LIM 3.2 or higher. The current context mapping is saved at entry and is then restored when QFIND terminates.

QFIND currently supports 512 directories and an unlimited amount of files.

QFIND can execute as a child process of a parent that uses EMS memory without interfering with the parents' data.

## Extended disk partition support

QFIND supports DOS disk partitions greater than 32MB. Note that QFIND will not work on some non-DOS partitions. A valid DOS boot sector is required on each drive that is searched. Use the DOS FDISK command to DISPLAY the partition table information when a non-DOS partition is suspected.

## Multiple drive search

QFIND will search multiple drives in ascending order starting with the drive specified by the user. If none is specified, then the first hard disk is assumed. Note that systems without a hard drive or RAM disk must ALWAYS specify the drive letter. QFIND defaults to the first drive after the last installed floppy disk.

## SNAPSHOT disk dump

The SNAPSHOT utility captures only the Boot sector, the FAT tables, and the ROOT directory. No user data is dumped. The captured data is then written to SNAPSHOT.DAT on the default drive. The utility does not dump SUBST, JOINed, NETWORK, or OS/2 drives. It utilizes Undocumented DOS Function 52h to determine the drive type and location of user data. DOS versions 3.1 and higher are supported. In debugging mode, QFIND operates on this disk image the same as it would on a real disk. The program can be single cycled to determine the exact nature of the problem.

## Fuzzy search

QFIND will support the following command-line arguments for embedded characters. Bracketing a string of characters with '*' tells the parser to match that sequence of characters anywhere in the filename and/or extension.

Rules:

1. A fuzzy search field must begin with the asterisk (*) in the first position of the name or file extension.
2. At least one valid DOS character must be between the two asterisk brackets. The '?' wildcard is valid. Two consecutive asterisks will return an error.
3. The fuzzy character count can be no more than 8 for the name field, or 3 for the extension field. This count does not include the brackets (*). An over length name or extension returns an error.

A full length field of fuzzy bytes will be interpreted as a non-fuzzy search, and no error will be returned.

| | | |
|---|---|---|
| IB*AB* | = IB?????? | .??? |
| *IB*.COM | = <<IB>> | .COM |
| *IB* | = <<IB>> | .??? |
| *IB*.i* | = <<IB>> | .i?? |
| *IB*.*i* | = <<IB>> | . <<i>> |
| *I?M*.COM | = <<I?M>> | .COM |
| *1234567*.*12* | = <<1234567>> | . <<12>> |
| *12345678* | = 12345678 | .???; Legal but useless |
| LOTUS.*123* | = LOTUS | .123; Legal but useless |

| | | |
|---|---|---|
| *123456789* | = ERROR | ; Over length name |
| **.Exe | = ERROR | ; Double splats together |
| LOTUS.*1234* | = ERROR | ; Over length |
| extent*A*B *C.EXE | = ERROR | ; More than 2 '*' wildcards |

## Command-Line Switches

See Registered Documentation for more information.

## Implementation

Screen performance has been improved by utilizing direct screen writes. Snow prevention is implemented only when the CGA card is detected. Snow checking is disabled when DesqView is active. QFIND uses IOCTL function 4408h to check each drive to determine if it is removable or fixed media. Removable media drives are read with BIOS, and fixed drives are read with DOS. This bypasses the DOS error message that is generated when the drive door is open. QFIND uses the DOS direct read interrupt Int25h. DOS 4 and higher versions deviate from earlier versions in the manner that this interrupt is implemented. QFIND automatically uses the extended partition calling convention for DOS 4 and up. QFIND operates correctly under MS-DOS 5.0 and the extended size disk partitions it supports. QFIND takes over one interrupt vector, Int23h (Control Break). The control break allows the user to quit QFIND at any point during the execution. The vector is restored upon any exit from the program. A user-entered ^C causes the program to terminate and restore the ^C vector at exit. QFIND performs direct reads of the disk and cannot damage the integrity of the disk. No writes are done to the disk. QFIND is written entirely in Turbo Assembler v3.2.

### PIF

QFIND is "ill behaved" in screen management functions as direct screen writes are used. QFIND is "ill behaved" for vectors as it takes over Int 23h, the Control-Break vector.

### DOS Errorlevel

0 - No errors

1 - Parsing error

2 - Insufficient Memory

3 - DOS read error

4 - Invalid boot sector

5 - User ^C

6 - Network was found

### CGA snow checking

CGA snow checking will be implemented when a successful test of the retrace hardware has been completed. Some oddball monographics cards do not correctly support the horizontal retrace bit in CGA status port 3DAh. The result is QFIND hanging forever, waiting for the bit to change state. The test first waits for the retrace bit to become clear, then waits for it to set again. If either test fails, then the retrace hardware is assumed defective and snow checking is disabled. It is also disabled when DesqView is active. Non-standard use of this port can cause unpredictable results. If this is suspected, the user may enable or disable CGA snow checking from the SET SWITCHES panel of QINST. This will override the start-up tests and force the CGA snow testing into the mode selected by the user. The suggested method is to set CGA snow checking to OFF which will bypass any potential hang condition. If no screen snow is encountered, leave the switch off. Performance is significantly improved with snow checking turned off.

### Error messages

```
Unable to locate or read header for.
```

The ZIP file is damaged or corrupted.

```
Logic error locating central directory
for
```

Internal error in QFIND

```
Insufficient memory to search this
subdirectory
```

A subdirectory greater than 64K is being searched and not enough memory is available to load it.

/L switch has requested a scroll buffer and no EMS memory is available. Scroll buffer removes 64K of DOS memory that is available for directory loads.

QFIND is trying to run in 384K or less.

QFIND is running from another program which has SHELL'd out to DOS.

```
Insufficient memory to search xxx.ZIP
```

Not enough free memory available to load the ZIP file central directory for searching.

```
Invalid boot sector for drive x:
```

Non-standard DOS boot sector was detected. Bytes per sector is zero or not contained in the set of 64,128,256,512,1024,2048,4096,8192.

Boot sector has returned Zero bytes per sector. Specialized RAM disks such as some AST products do not utilize a standard DOS boot sector.

## Usage

Valid switches are /G /H /L /P /S /V /Z

/G Goto Match

The default drive and directory are changed to that of the first matching file.

/H Hidden file(s)

Only file attributes that match either the system or hidden bits will be displayed. This switch disables the /Z switch.

/L List scroll buffer

The first 819 lines of output are saved in a scrollable buffer. This switch disables the /P switch. I/O redirection cancels this switch.

/P Pause mode

The program will pause when the screen is full.

/S Single drive

Only a single drive will be searched.

If not specified, the default drive is assumed. Otherwise, the first file specified by the user is the only drive searched.

/V Verbose path display

Every path searched will be displayed. Useful to build a complete path table of the specified drive(s) by searching for a nonexistent file.

/Z search zip files

All PKZIP files will be searched for member(s) matching the input argument. Corrupted ZIP files will be identified with an error message. Empty ZIP files are not reported.

## Command-line examples

```
QFIND file.
```

Start with the default hard drive.

No dot defaults to all file extensions.

Match all occurrences of 'file????.???'

```
QFIND file
```

Start with the default hard drive.

The dot specifies no file extension.

Confine match to 'file'.

```
QFIND *
```

Start with the default hard drive.

Find all files.

```
QFIND *.*
```

Start with the default hard drive.

Find all files.

```
QFIND *.* /H
```

Start with the default hard drive.

Find hidden files only.

```
QFIND *.* /V
```

Start with the default hard drive.

Display all paths as they are scanned.

```
QFIND *.* /H /S
```

Start with the default drive.

Find hidden files only.

Check only the default drive.

```
QFIND A:*.*
```

Start with the A: drive.

Find all files.

```
QFIND G:*.* /S
```

Start with the G: drive.

Find all files.

Check only the G: drive.

```
QFIND *.* /H /S > PRN
```

Start with the default hard drive.

Find hidden files only.

Check only the first hard drive.

Redirect output to printer.

```
QFIND *.* /H /S > C:\Temp\Dir.Lst
```

Start with the default hard drive.

Find hidden files only.

Check only the first hard drive.

Redirect output to disk file on drive C:

```
QFIND *IBM*
```

Find any occurrence of the letters 'IBM' in the file name field. The file extent field defaults to .???

```
QFIND *IB??IO*
```

Find any occurrence of the letters 'IB??IO' in the filename field. The file extent field defaults to '.???'.

This example would find IBMBIO.COM.

```
QFIND *.*A*
```

Find any occurrence of the letter 'A' in the file extension field. The letter can be in position 1, 2, or 3. The filename field defaults to '????????'.

```
QFIND E:Command.Com /G
```

Find COMMAND.COM starting with the E: drive and change to the drive:directory where it was found.

## SUBST drives

DOS allows disk subdirectories to appear as a different drive letter. This is done by using the SUBST command.

```
SUBST E: C:\UTIL
```

will cause the C:\UTIL subdirectory to appear as Drive E:.

QFIND and many DOS commands will ignore a SUBST drive. Under DOS 5.0, the following commands will not work, or are not recommended with a SUBST drive:

Assign
Backup
Chkdsk
Diskcomp
Diskcopy
FDisk
Format
Label
Mirror
Recover
Restore
Sys

## Networks and OS/2

The current version QFIND does not support network drives or OS/2. It will complain and exit if either is detected. OS/2 is identified by checking the DOS version. A major value of 10 (OS/2 v1.0) or higher is returned by OS/2.

## DesqView

QFIND is DesqView aware. It will make a call to DesqView to determine the location of the DV Alternate Screen Buffer. This display buffer will be used for all screen writes.

Cursor control and keyboard routines are done entirely with BIOS. This enables QFIND to run in the background in a small window. No polling is done during scroll buffer viewing. This allows DesqView to swap out QFIND when it is waiting for user input.

The video display routines are coded as 'DesqView Critical.' The message display routine cannot be swapped out until it has completed its function. Other than the color banners, a message display is a single line of text. QFIND crashes with old versions of QEMM (v5.00).

## BUG Reports

Bug reports are welcome.

1. Make/Version of DOS
2. Copy of AUTOEXEC.BAT and CONFIG.SYS
3. Names of TSRs if any, that are loaded
4. Redirected copy of QFIND in verbose mode, e.g., QFIND<nonexistent.file> /V > DUMP.FIL. This provides a list of all subdirectories.
5. Use SNAPSHOT to take a dump of the suspect disk.
6. Run CHKDSK against the failing drive.

Attempt to diagnose any bugs by unloading all TSRs and start with as simple a system as possible.

## QFIND v2.22 Registration Form

Name: ______________________________

Company: ___________________________

Address: ____________________________

City: _______________________________

State: __________ Country: ____________

Zip: __________ Phone: ______________

Your EMAIL: _________________________

QFIND found on: ______________________

OLR Password: ______________ 3 to 8 chars

**Remit to:**

Bruce Gavin
9341 Ridgeside Lane
Orangevale, CA 95662

CompuServe: 70137,3244
Online Reference BBS:
(916) 988-7551 (voice)
(916) 988-0920 (data) 9600 v.42bis / 8N1

Diskette deliveries come with additional Online Reference shareware files.

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Total Copies x Unit Price .......................... $______

Total enclosed ........................................ $______

**Terms:**
Check or Money Order drawn on a USA bank in U.S. funds.

**Upgrade Fees for Registered Owners:**
Diskette Upgrade $5
Overseas Diskette Upgrade $7
Unlimited Upgrades via BBS No Charge

**Registration Fees:**
BBS delivery $20
Diskette delivery $25
Overseas diskette delivery $27

Please see the file ORDER.FRM for multiple-copy discounts.

Site license provides one diskette to create multiple copies. Please include your EMAIL address on the order form.

This allows you to be easily notified for updates.

# RELDEL

**Version 1.37s**
**The Real File Deleter**
**Copyright © 1990–1993 by TARDIS DP Consultants**

## Introduction

RELDEL was born out of safety and security concerns with the standard DOS DELETE command. The DOS command has two potential problems which we needed to work around. First is safety. The DOS DELETE command automatically deletes all the files matching the file specification without prompting. If you accidentally type in DEL *.BAS instead of *.BAK, you have deleted your Basic source code instead of backup files. Second, even if you delete a file, many common utilities and the DOS UNDELETE command can restore the file to predeleted condition. If you are deleting sensitive information, and you don't want anyone coming behind you and resurrecting your file, the DOS DELETE command is not for you.

RELDEL, on the other hand, automatically prompts you to delete each file. You must respond **Y** before the file is deleted. Note that there are command-line parameters which will delete files without prompting. RELDEL also destroys the contents of any file (even read-only, system, or hidden files) before deleting them. After writing random ASCII garbage, RELDEL writes out all ASCII 0 and then truncates the file to 0 bytes. Finally, the file is deleted. It is impossible for any utility to resurrect the file in usable condition.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the RELDEL files in the directory \SECRETS\RELDEL. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the RELDEL files to a directory that is included in the search path.

## Using RELDEL

RELDEL is incredibly easy to use. You have three options for its use:

1) Display each file on the screen and prompt for its deletion.
2) Display each file on the screen and automatically delete it. No prompting is performed.
3) Display nothing on the screen and don't prompt for anything.

## Options

Here is a description of each option and its parameters.

### Display each file on the screen and prompt for its deletion.

The command syntax is simply RELDEL <filespec>. For example, if you type

```
RELDEL *.TXT
```

all files with the TXT extension are permanently deleted. No command-line options should be entered.

### Display each file on the screen and automatically delete it. No prompting is performed.

The command syntax is similar to the above example except that you add a space and /P to the end of the command. For example,

```
RELDEL F*.* /P
```

will delete all files which start with the letter F, but will not prompt you for permission to delete them. Each file still displays as it is deleted.

### Display nothing on the screen and don't prompt for anything.

The command syntax is similar to the above example except that you add /D to the end of the command. For example,

```
RELDEL *.DBF /P/D
```

will delete all files with the extension DBF but will not prompt you for permission to delete them. Absolutely nothing is displayed on the screen during the process. This option is very handy in batch files or as called from another program.

As you can see, RELDEL is both powerful and easy to use. If you ever need help with the command syntax, you may simply enter RELDEL with no file specification. Help information automatically displays.

***Note:*** As a safety precaution, RELDEL should only be used to delete files in the current directory.

### RELDEL 1.37s Registration Form

Name:
Company:
Address:
City:
State: Country:
Zip: Phone:

**Remit to:**
TARDIS DP Consultants
Department 3
6 Sedely Ct.
Greensboro, NC 27455

To register RELDEL, send a check or money order for $7 (payable to Charles L. Cranford, the author).

# SeekEasy

**Version 7.07**

**Copyright © 1992 by Correlation Systems**

## Overview

SeekEasy is an easy-to-use *fuzzy-logic* text-finding program. It scans your disk files, finding text that even approximately matches what you asked for and displays what it finds in *best-matches-shown-first* order. It neatly solves those "I *know* it's in here *somewhere*!..." situations.

With SeekEasy, you don't have to remember the exact wording or spelling of the text in the file you're trying to find. If your requested text is even *close*, SeekEasy will find the file! For example, if a file somewhere on your disk holds the text "...THE TRUNK OF THE ELEPHANT IS...", SeekEasy can find it even if your request is for "EPHALANT'S TRUNQ."

Typical applications: finding all your source code files — perhaps created over many years — that reference a function name you now wish to change; finding the filename of a year-old letter to your Uncle Mort when you don't remember exactly the name you used in the letter, etc.

SeekEasy can search through one file or up to all the files on an entire drive. Matched text is displayed along with its filename. The *probable match* words are highlighted and twelve lines of the file around the matched words shown for context. The display window may be scrolled forward and backward through the file to show more surrounding text.

SeekEasy will scan *all* file types — word processor, database, even **.COM** and **.EXE** files if desired. It can be told which filenames or file types to ignore and/or to always include in the search. It has many default settings of this type that are user-customizable through simple menus.

***Note:*** SeekEasy will *find* text, but it won't *alter* it — you must use the appropriate word processor, database program, etc., for that.

## Some Detailed Features Of SeekEasy

- It will handle files of any length. It displays 12 lines of a file on the screen at once, and you can scroll the 12-line window through the entire file if you wish one line or a screen at a time.
- If so instructed, it will search one file or all the files in a directory, a directory and all its *child* directories, or a whole drive's worth of files. It will also search through specific files on a list and/or skip files on another list you can set up within SeekEasy.

  For example, SeekEasy can be set to automatically search, say, REFERENC.TXT, plus all *.LST files, while skipping over a particular file named BIGFILE.LST. And these choices can be saved as the *default* mode of operation.
- SeekEasy will search floppy disks, hard disks, Bernoulli-Box drives, etc.
- At the end of a search, SeekEasy shows you the items it found in *best-matches-first* order — the highest scoring (as evaluated by its fuzzy-logic matching algorithms) block of text is shown first, then the next-highest scoring, etc.
- You can mark blocks of text that SeekEasy has found, then send the blocks of text out to a printer or a disk file for reviewing later.
- SeekEasy is highly customizable. Defaults can be set specifying the files to search, search mode, turning beeps on or off, printer port to use, screen colors, etc. — all through simple menu choices.
- SeekEasy does not use up valuable disk space with index files, as do cross-referencing programs.
- Wherever it makes sense, each screen in SeekEasy has one or more pages of Help Screens behind it, available at the press of the F1 key.

## System Requirements

SeekEasy will run fine on just about any IBM PC or compatible computer. The minimum system requirements are

PC-DOS or MS-DOS, version 2.0 or higher.

If running under Windows, Windows 3.0, or higher.

256K or more of RAM.

One or more disk drives.

Monochrome or color monitor.

Parallel- or serial-interface printer for printouts.

## Installing SeekEasy

General issues:

There are two things you should probably do before you install and use SeekEasy. Neither is required, but each may make using SeekEasy easier and more productive:

A) Run the DOS TREE command and direct the output to your printer, for later study. This will show you how your disk is divided up. To do this at the DOS prompt, type **TREE > PRN** and then press Enter. This assumes your PRN device is on, and if it is a serial printer, it has been set to the proper baud rate.

B) If it's not already part of your AUTOEXEC.BAT file, issue the following command at the DOS prompt: **PROMPT $P$G** and press Enter. This changes the standard DOS prompt to show your current location in the directory tree. It would be best to place this command into your AUTOEXEC.BAT file, so it will be invoked each time you turn on your system.

SeekEasy will run under Windows 3.0 and above. It operates the same as under DOS, except that under Windows it can perform searches in the background (in 386 Enhanced Mode).

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the SEEKEASY files in the directory \SECRETS\SEEKEASY. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the SEEKEASY files to a directory that is included in the search path.

For best performance, copy both the SEEKEASY.EXE and the SEEKEASY.PIF files from our disk to your \WINDOWS directory. An alternate choice would be to copy the two files to any directory on your DOS *Path* list.

## Running SeekEasy

### Running SeekEasy under MS-DOS

To run SeekEasy once it's installed per the above instructions, just type **SEEKEASY** and then press Enter. If the program does not run, it is probably because it is not in a directory on your DOS Path.

If you wish to run SeekEasy under the DOS 5.0 DOSSHELL program-switcher, we refer you to your DOS manual. See Chapter 3 (MS-DOS Shell Basics) and Chapter 8 (Customizing MS-DOS Shell). They explain it better than we could.

### Running SeekEasy under Windows

Since it's basically a DOS application, SeekEasy runs best in the full-screen mode. It will do searches in the background, if Windows is running in the 386 Enhanced mode.

In the preferred full-screen mode, you may use the standard Windows program switching keys: Alt-Enter toggles SeekEasy between a window and full-screen display mode. Alt-spacebar brings up the "switch to another program" menu, and Alt-Esc toggles between SeekEasy and any other open program windows.

The SEEKEASY.PIF file is a Program Information File that tells Windows about the program and its capabilities. SeekEasy will run under Windows just using the default PIF Windows assigns to unknown programs, but it will run better if Windows can use the customized SeekEasy .PIF file.

### Problems to watch for when running under Windows 3.0

To search files in a CD-ROM under Windows using SeekEasy or any other program, two things have to be verified or done:

A) If it isn't there already, add the following statement to your SYSTEM.INI file in the [386Enh] section:

```
DEVICE=LANMAN10.386
```

Yes, you may not be on a LAN (Local Area Network), but you still have to do this, say the Windows folks, in their README.TXT file. Believe it. If you don't do this, you WILL eventually see crashes while searching your CD-ROM with SeekEasy!

B) If it isn't already there, expand/copy the LANMAN10.386 file from the Windows source disk to your C:\Windows directory. Note that you have to *Expand* this file while copying it. It is delivered in compressed form on the Windows source (floppy) disk. For Windows 3.0, the compressed version is 1548 bytes long and the expanded version is 8786 bytes. To expand the LANMAN10.386 file, first find the EXPAND.EXE program file on the Windows source disks, and copy it into your hard disk's \Windows directory. Then insert the Windows source disk that has the LANMAN10.386 file on it, and from your Windows directory on your hard disk issue the command:

```
EXPAND A:LANMAN10.386 C:\WINDOWS
```

You should then see the LANMAN10.386 file on your hard disk, and it should be larger than the copy on your Windows source (floppy) disk was.

# Screens

SeekEasy uses a variety of screens.

## Main search screen

To perform a search, you need to specify two things:

A) The text you'd like to find matches to

B) Which file or files you want the program to search

Pressing the Enter key toggles you between the two fields that specify these items.

You specify the text to be matched in the blank screen area to the right of the **Search for:** prompt.

Specifying which file or files to search is a bit more involved. The basic file specification is the file-spec indicated in the blank screen area to the right of the **File Spec:** prompt. This basic specification can be added to, subtracted from, or overridden by settings made on the **Search Spec** screen which can be reached by pressing F2. See the discussion in the following section regarding the Search Specification Screen for a more complete description of how this all works.

Note that when you press Enter to select the **File Spec:** area, a box pops up showing **Search Specification** information. This is just a summary of the major settings from the Search Specification Screen, shown for your verification prior to your starting a search. To actually change the settings, you must press F2 to go the Search Specification Screen.

The file specification may include the DOS wildcards ? and *. ? means *any single character matches this*, while the * means *any one or more characters matches this, from here to the end of the filename prefix or suffix.* See your DOS manual for a more thorough description of wildcards. The file specification most people use is just *.*.

When you're viewing the results of a search, you may occasionally notice that the same line of information appears in two different items. This is normal. In *Stream* mode, to ensure the text you are looking for is not split — partially in one item and partially in another — the items overlap by one line. Thus, if a search found an item with high-scoring text in the last line of the item, it would also see that the next item (with its one line of overlap) STARTED with that line, and give it a high match score also.

In *Formatted* mode, the program logic tries to break *items* at the ends of sentences. If it cannot find what it believes is the end of a sentence within the last three lines of a screen's worth of text, it will arbitrarily force the end of the item, and set a one line duplication/overlap as is done by the Stream mode. Except for this case, *Formatted* mode does NOT overlap item boundaries.

If you have set the settings on the Search Specification screen to perform a search that includes the files listed on the *Scan List* as well as the one you specify on the Main Search Screen, then the instructions to the right of the *Search Spec:* prompt will change to indicate that putting an entry there is optional. The program will start a search without any specified file(s) here since the Scan List entries on the Search Specification Screen will satisfy the requirement that *some* file specification must exist before a search can be done.

Due to memory limitations, SeekEasy will display only the 200 best matches to the text you told it to search for. If what you ask for isn't within the best 200 matches, you need a clearer search specification!

SeekEasy DOES "evaluate" all of the matches it finds while searching. The 200 items you see after a search operation are the BEST 200 it could find, not just the first 200!

When you press F10 to begin a Search operation, SeekEasy may occasionally warn you that it appears you are performing the same search you just did, and ask for verification. The purpose for this is if you had just completed a LONG search of a big hard disk or CD-ROM, you DON'T want to accidentally press F10 again, throw away all the search results, and start the search all over again! SeekEasy therefore requests confirmation before it starts the search, to ensure that you really *do* want it to "forget" all the search results but repeat the identical search again.

Starting a Search Operation erases all records of the previous Search Operation, so be sure you are done with the results of one search before beginning another.

Strange looking characters on the screen, unusual data display, etc.: SeekEasy attempts to work with the widest possible range of data formats. One of the ways it does this is to clip off the high bit (bit 7, counting from 0) of the file data it reads. This allows files in WordStar and similar formats to be read without confusion. This bit is normally 0 for standard ASCII so removing it creates no problems. However, for graphics characters, clipping off this bit sometimes converts the graphics character into a printable character, which will be displayed on the screen. This is normal and just a function of SeekEasy's efforts to convert everything it sees into human readable format.

Also note that while Formatted mode tries to display data in a more normal manner — expanding tabs, recognizing carriage-return/line-feeds, etc. — it also clips the high bit and will also generate the same strange looking characters.

Down near the bottom right corner of the screen you'll see two status items displayed — **MODE:** and **BEEP:**. These are information-only displays of settings made in the Utilities Settings screen, reachable by the F3 key. To change these, you have to go to the Utility Settings screen. These status items are shown here just for your information, as it is often useful to know what they are prior to starting a search. The **BEEP:** setting can also be changed DURING a search.

### Search specification screen

Any changes you make to the settings on this screen only apply for the current SeekEasy session, unless you save them as your new default setting. See the F10 discussion below.

The discussion that follows assumes you are familiar with DOS's directory structure. If not, please review the information in your DOS manual regarding directories, paths, and so on.

Once SeekEasy is running, it ignores any DOS path settings. SeekEasy's search area is controlled only by the path information you specify on the Search Specification screen.

Starting at the top of this screen, and working down:

The *current location* shown just below the screen title refers to the drive and disk path/directory that you were in when you started the SeekEasy program. Once SeekEasy has been started, this cannot be changed. These settings are the ones used in the F2 operation discussed below, if no explicit settings are shown there.

**F2 — START SEARCH AT**: This specifies where SeekEasy BEGINS its search. It may also be where the search ends, depending on other settings on this screen.

A typical entry in this section would be **C:\TEST**. In this, the DRIVE (or disk) is specified as **C:** and the STARTING DIRECTORY is specified as **\TEST**.

This area is left blank in the "as shipped" configuration of SeekEasy. With this blank, when you start SeekEasy from a given directory, the search automatically starts in that directory, on that drive. This was deemed the most general-purpose setting possible.

Note that one common mistake is to put an entry in this spot of **C:**. This specifies the DRIVE (**C:**) but NOT the **root directory** (\).

**F3 — SEARCH AREA**: This specifies HOW FAR the search goes once it is started in the drive/directory set by the F2 setting above. The options are **this directory**, **this plus child directories**, and **the entire drive**.

The first choice — *just the selected directory* — says to search all the files in this directory that meet the file spec criteria.

The second choice — *child directories* — says to include all subdirectories in the search.

The third choice — *the whole disk option* — says only to use the drive information ("C:") in the F2 setting. Any directory/path information is ignored, since the search will always start at the root directory of the specified drive and cover ALL directories in the drive.

**Example:**

Suppose you had a disk directory structure (simplified for this discussion) as follows:

```
C:\
 | |—DOS
 | |—UTIL
 | |—TEST
            | |—HARDWARE
            | |—SOFTWARE
```

Further, suppose your *starting directory* was **C:\TEST**.

- If you chose **just the selected directory**, your search would only cover the files in the C:\TEST directory.
- If you chose **selected directory plus its child directories**, then your search operation would cover the files in THREE directories:

  C:\TEST
  C:\TEST\HARDWARE
  C:\TEST\SOFTWARE

- And, if you chose the **whole disk option**, search would cover ALL SIX directories — the three above, plus:

  C:\ (the root directory)
  C:\DOS
  C:\UTIL

**F4 — SEARCH FILE**: While the F2 and F3 keys above specify the area(s) on the disk to be searched, this specifies which files within those areas will be opened and examined.

SeekEasy builds a *file specification* — or file spec for short — from up to three sources:

A) The **File Spec:** you entered on the Main Search Screen.
B) The Scan List file specifications.
C) The Skip List file specifications.

This entry controls which of the above three sources are active during a search.

There are four choices:

- **The Specified File Only**: This tells SeekEasy to JUST search the file(s) specified in the **File Spec:** entry the user typed in on the Main Search Screen. You may have specified a specific file (**MYFILE.DOC**) or a group of files, using wildcard characters (**MYFILE.***).
- **The Specified File Blocked By The Skip List Names**: This is the default configuration as shipped. This tells SeekEasy to operate the same as in the above mode, EXCEPT to ignore any file whose name is covered by the filenames (or specifications using wildcards) shown in the Skip

List. This is normally used to speed searches by having SeekEasy skip **.EXE** and **.COM** files, etc.

There are two things to know about priorities here:

First, the Skip List can block a file specified by the **File Spec:** entry. In other words, if a user specified a file spec of **HELP.***, and had a Skip List entry of ***.EXE**, SeekEasy would search a file called HELP.DOC, but would NOT search a file named HELP.EXE, if it existed.

Second, the Skip List will NOT block a user-entered **File Spec:** entry if that entry is SPECIFIC — if it has no wildcards in it. Thus, if in the above example the user had specified a file spec — not of **HELP.*** but of **HELP.EXE** — then SeekEasy WOULD search the HELP.EXE file, even though it was on the Skip List.

- **The Specified File, Blocked By The Skip List Names Plus The Scan List Names**: This acts just like the option above, except IN ADDITION TO the user-entered **File Spec:** name, the program also uses all of the entries, if any, in the Scan List.

  As before, the Skip List can block Scan List or user-entered **File Spec:** entries, unless they are specific filenames (without wildcards).

  If this option is chosen, the user does not have to actually enter a **File Spec:** entry on the Main Search Screen to start a search, as long as the Scan List has at least one entry.

- **The Specified File, Plus The Scan List Names**: This acts just like the option above would act if the Skip List were empty: it searches **File Spec:** plus Scan List files.

  As an example: If you wished to search, say, all **.DOC** and **.TXT** files, you could specify this any number of ways, all equivalent. Here are two:

  A) Enter ***.DOC** for the **File Spec:** info on the Main Search Screen, and put ***.TXT** in the Scan List, then choose the Search Files choice of Specified File Plus Scan List Names.

  B) Or, you could leave the **File Spec:** entry on the Main Search Screen blank, and put ***.DOC** and ***.TXT** both into the Scan List, and enable that.

**F5 — SKIP LIST**: This is used per the above instructions. Entries must be separated by spaces. They can be uppercase or lowercase, or any mixture. Note that if you specified a file specification of ***.*** here, it would block ALL files from being searched when this list is activated, except for any file specified on the Main Search Screen without any wildcards at all. You normally should NOT use a file specification of ***.*** here.

The normal use for the Skip List is to ignore files you don't want to waste time searching through. Users normally customize this list to block the scanning of large files on their systems that have nothing of interest in them. The settings shipped as the factory defaults are only a first approximation of the files most users want to skip. You should change these settings to suit your preferences, then save the settings as the new default settings.

**F6 — SCAN LIST**: Same formatting, etc., as the above. You MAY use ***.*** here if you wish, without any danger of missing files.

**F10 — SAVE AS NEW DEFAULTS**: This takes ALL of the settings on this screen, plus those on the **UTILITY SETTINGS** screen, and saves them to disk as the new default values. The next time SeekEasy is run from disk it will come up with the settings as they were when you saved them as your new defaults.

Note that the defaults settings are stored inside the SEEKEASY.EXE program disk file. There are no separate configuration files to worry about. This will become apparent by the status messages issued during this operation, as the program finds the SEEKEASY.EXE file, then finds the proper spot to insert the information, then successfully saves it. However, you will NOT cause any problems if you rename SeekEasy to something shorter such as SE.EXE, etc. It "knows" what name it's run as.

## Utility settings screen

As in the Search Specification screen, any changes you make here only hold for the current SeekEasy session, until you save them as your new default settings, see the F10 discussion below.

Going down the functions, from the top of the screen on down:

**F2 — BEEP AT END OF A SEARCH**: This one's pretty clear. Try the three choices and see which one you like. Note that this item can also be changed DURING a search operation itself — see the bottom of the *Search In Progress* screen for what key to use. A change made then is just like a change made here. It will remain as you set it, but will not become the new default setting until you save it to the disk as such.

**F3 — BEEP ON ERRORS**: This either enables or not a single beep whenever an error occurs such as disk-read error, trying to send a marked block out to a printing device that doesn't exist, etc. Set it to match your preference.

**F4 — EXPAND TABS:** The program help screen says it all.

**F5 — PRINTER LOCATION**: Used to select the destination of marked blocks sent to the printer. The factory default setting is LPT1. ***Note:*** If you select a non-existent printer, or if you have a printer problem while printing marked blocks, don't panic. Even if everything appears locked up, be patient. At the most, in about 90 seconds SeekEasy will unlock things and recover, giving you an error message explaining the problem.

If you're using a serial printer and it doesn't respond, suspect baud rate settings, or that the printer or its cable is not wired to hold both the CTS and DSR signals high when it's ready to accept data.

**F6 — PRINTER LEFT-MARGIN OFFSET**: The program help screen says it all. Experiment with this to see what your printer is most happy with, then save it as the new default setting.

**F7 — SEARCH AND DISPLAY MODE**: The program's help screen pretty much covers this. ***Note:*** If you do any *block mark* operations, then change this mode from Formatted to Stream or Stream to Formatted. The blocks you marked may now have their start/finish points in error by one line or so.

**F8 — FLAGGING MATCHED TEXT IN REVERSE VIDEO**: This applies to the results of Search operations. Words that the SeekEasy algorithm thinks are significant are normally flagged in reverse video to make the probable matches easier to find on the screen. How the displaying is done does not change how items are evaluated or scored, only which words are flagged in reverse video.

The choices are

- **Flag All Matches**: SeekEasy flags any word fragment that it thinks is even a little bit significant. This gives you the most information, but results in a very cluttered display.
- **Flag Medium And Strong Matches**: This raises the threshold of what should be shown in reverse-video slightly, to get rid of some of the clutter. This is the factory default setting.
- **Flag Only Strong Matches**: This only reverse-video flags good, solid matches but may cause you to miss something "sorta similar" to what you asked for because it wasn't flagged.
- **Flag No Matches At All**: If you don't like the reverse-video flagging of probable matches, choose this — it doesn't flag anything in search results, just shows you what it found in the standard best-matches-first order.

***Note:*** This setting can be changed at any time without affecting the search results. You can look at the search results, go to this screen, and change this setting, then return to the Main Search Screen and see how the data looks at the new setting.

**F9 — TEXT/BACKGROUND COLOR SELECT**: The factory default setting for this is #3, yellow/blue. When it's started, SeekEasy checks the hardware configuration and either allows color choices or allows only black and white choices. You can change the choices, and even save them as the new defaults, but SeekEasy will still allow only what it thinks are the proper range of choices the next time it is run. You can force it to think it is running on a color or black-and-white system by invoking SeekEasy with an extra parameter, as

```
SEEKEASY C <Enter>
SEEKEASY M <Enter>
```

**SEEKEASY C** forces the program to ignore the hardware configuration it detects and allow color choices.

**SEEKEASY M** tells it to act as if it detects a black-and-white (monochrome) system and allow only B/W color combinations.

**F10 — SAVE DEFAULTS**: See the description of the similar key in the description of the Search Specifications screen. Note that if you do this, it saves both the settings on this screen AND the settings on the Search Specifications screen, as the new defaults.

### Block mark/print screen

The same scrolling keys work essentially the same way here as they did in the Main Search Screen.

Follow the screen instructions to mark the block. The text shown in reverse-video is what will be sent to the printer or disk file you specify.

Be sure to move the cursor to the line PAST the last line you want included in your block.

If you choose to send the block to a disk file, SeekEasy may tell you that the file that you specify already exists. If so, it will give you the option of overwriting the file or appending the block onto the end of the file. Overwriting erases the old file and replaces it with a new one holding just the block you've marked, while Appending simply adds the content of the block onto the end of the existing file.

When the marked block is being sent to the printer or a disk file, the display will position the text such that the top line in the display area is the one currently being sent to the printer or disk file. Thus you can monitor the progress of the operation by watching the display and seeing what data is currently going out. At the end of the operation, the last line of the marked block will be shown at the top line of the display area.

### Searching screen

You may end the search operation at any time by pressing the ESC key. SeekEasy will use whatever it has managed to find in its search up till that time.

Each *progress indicator dot* indicates 10% of the file has been read.

**F3** : During a search (but not after), if you realize you wish to change how or if the computer beeps at the end of a search, just press F3 to skip through all the possible *beep after* settings. Just press the key once, then wait — it may take a second or so for the program to break into its search at a convenient spot to act on your keypress. You'll see the new status reflected at the top of the screen.

**F9** : If you see that the program has started scanning through a large file, and you know that the file doesn't have any information in it you want, you can skip the rest of it by pressing this key. SeekEasy will go on to the next file and continue its search.

If the file (or file type — .PIC, whatever) you skipped is a file or file type you regularly have to skip over, you should consider adding it to the Skip List and saving that as your new default setting.

## Performance Tips

See the full manual supplied to registered users for information regarding how to maximize the speed of your Search operations. There are many (sometimes interacting) choices in this area.

The full manual also contains information describing how the SeekEasy algorithm "sees" your search request. This will help you perform more accurate searches, increasing the odds that the item you want will be the first one or two SeekEasy shows you after a search.

## Technical Support

Correlation Systems offers unlimited telephone and by-mail support of SeekEasy to registered users. We'll answer simple questions about getting SeekEasy up and running from anyone, but we can handle more complicated issues only for registered users.

## SeekEasy 7.07 Registration Form

Name: ____________________
Company: ____________________
Address: ____________________
City: ____________________
State: ________ Country: ____________
Zip: ________ Phone: ____________

**Remit to:**
Correlation Systems
P.O. Box 39
Lomita, CA 90717
(310) 833-3462,

Send cash, check, or money-order (no credit card orders).

SeekEasy is distributed as SHAREWARE. The full registration fee is just $39.95. For this you receive a disk with the latest SeekEasy program version, a full user's manual — including many "performance optimizing" tips available nowhere else, and unlimited telephone support for SeekEasy-related questions. Also, as a registered user you are notified about upgrades and receive special low pricing.

TO ORDER: Please fill out the form, and include $39.95 (California residents also add sales tax; for orders from outside the USA, add $10). All payment must be in U.S. funds, payable on a U.S. bank. Prices already include postage and handling. If you have any questions, call 10 a.m.-6 a.m., California time.

Amount each ................................ $39.95 $ ______
Sales tax (Calif. residents only) ................ $ ______
Orders from outside USA add ........ $10.00 $ ______
Total enclosed ........................................ $ ______

Payment by: ☐ Check ☐ Cash ☐ Money Order

Specify Disk Size: ☐ 5.25" ☐ 3.5"

Lot # 0007

# SPINTEST & SPINTIME

**by Steven Gibson**

## Introduction

Please see "A GUIDE TO HARD DISK SECTOR INTERLEAVING" included in the file INTLEAVE.DOC. A few excerpts are included here. The guide carefully shows how to use the two included programs to determine whether your own IBM PC or compatible's hard disk drives have their sector interleave set correctly. The too often neglected SECTOR INTERLEAVING factor of a hard disk has a dramatic impact on data transfer rates. So many computers are so badly interleaved that it's quite likely that you could increase your own hard disk's performance by four to seven times just by optimally setting your disk's interleave factor!

Included with this INTLEAVE.DOC file are two small programs: SPINTEST.COM and SPINTIME.COM. The main program, SPINTEST.COM, determines how many disk revolutions your IBM compatible personal computer requires to read an entire 17 sector track from its hard disk. (SPINTEST is also fully compatible with the 25 or 26 sector tracks used with the high-density RLL controllers.) Running SPINTEST on the many computers at Gibson Research, and in many local computer dealerships, we discovered something quite surprising: Most computers being sold today are horribly interleaved. We turned up many machines which required more than 17 entire revolutions just to read or write one full track! These same machines flew along at 4:1 but not at 3:1. Any way you look at it, the issue of sector interleaving involves the complex interactions among the disk drive, the controller make and model (even the revision level), the processor, and system clock rate.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the "SPIN" files in the directory \SECRETS\SPINTOOL. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the "SPIN" files to a directory that is included in the search path.

## Overview of the Interleave Diagnostic Programs

The issues involving hard disk sector interleaving have gone unnoticed for so long because there has never been a clear way to see what's really going on deep inside a hard disk. After seeing the importance of this issue, we decided to change this. The two programs SPINTEST and SPINTIME determine exactly how many revolutions the disk requires to transfer one entire track of data for any standard MFM or RLL controller. The programs were handwritten in 100 percent machine language (as is everything Steve Gibson writes and Gibson Research publishes) to give them the required measuring resolution speed.

## The Spintest Program

SPINTEST determines exactly how many revolutions your hard disk currently requires to perform a full track transfer. DOS transfers a full track whenever programs or files larger than 9K are read or written, which is most of the time. SPINTEST does NOT directly read your drive's interleave, but the drive's interleaving performance can be easily inferred from SPINTEST's full track transfer revolution count.

SPINTEST averages the time required for each of over two hundred full track reads in order to accurately determine the revolution count per read. Then, the number of revolutions required to read just one track and the controller's full-track data transfer rate are computed. SPINTEST only reads data from your drive so data is NEVER altered.

Over two hundred track reads were used because of the inconsistent disk transfer behavior of AT&T's 6300 machines. These exhibit a maddening inconsistency in their ability to transfer disk data. Sometimes they get the next sector and often not. This means that a much looser than normal interleave generates MUCH BETTER overall disk throughput for AT&Ts. For this reason, and to guarantee correct results on any machines which may behave similarly, SPINTEST performs many track reads and averages the results.

Since SPINTEST measures track read time, it must assume a given ratio between elapsed time and disk rotation rate. Some "weirdo" clone computers have a clock which runs faster than normal when in their "turbo" (8 MHz) mode, and some lap computers have tiny hard disks spinning at weird speeds. If these do not sound like your situation, SPINTEST will deliver correct readings and you'll not need SPINTIME's confirming measurements.

But if EITHER of these cases might be you (if you have a weirdo clone computer or hard disk sporting laptop), SPINTIME will tell you for sure. SPINTEST is only usable when SPINTIME gives standard readings. The single (not surprising) exception to this is for the AT&T 6300 and 6300 Plus machines. Due to the overall problems they experience with disk transfers, SPINTIME may show a reading lower than 3600 RPM. For 6300's this is normal. SPINTIME is interesting regardless since it determines EXACTLY how fast your hard disk is really spinning!

## Running SPINTEST

To run SPINTEST, simply type its name at the DOS prompt.

***Important!:*** Read the section IMPORTANT SPINTEST AND SPINTIME NOTES, before you begin!

Since SPINTEST takes the average of over 200 track reads, the time required to run this test will vary between approximately 14 seconds for a fast 2:1 interleaved machine to 98 seconds for an incorrectly interleaved 26-sector RLL encoded hard disk. So simply type **SPINTEST** at the DOS prompt and wait a minute or two.

## Running SPINTIME

SPINTIME has two purposes: To determine the speed of your system's clock (if you suspect that it might be weird), and to verify that your system's hard disk is spinning at close to the 3600 RPM standard. Your system's clock is checked by measuring SPINTIME's exact execution time. It should require EXACTLY 60 SECONDS to run, no matter what. During that time, it is busy watching your hard disk spin, counting every revolution. After EXACTLY 60 seconds, it displays your drive's exact RPM (within its measuring resolution.) IF SPINTIME's total execution time is NOT EXACTLY 60 seconds or the displayed RPM is not close to 3600, SPINTEST's reported revolution count won't mean anything either. SPINTIME's real value is to assure you that SPINTEST is delivering worthwhile answers.

## Important SPINTEST and SPINTIME Notes!

Both programs assume and require DOS 2.0 or higher. They operate on your system's FIRST hard disk only (drive C). The hard disk must be bootable, and have its controller plugged into a slot that does not require special CONFIG.SYS device drivers for operation. SPINTEST and SPINTIME will never disturb your hard disk data, but if your disk is some kind of odd-ball, they may not function correctly.

The tests will be more accurate if your normal collection of memory-resident programs (if any) are not in memory at the time. Many resident programs "steal" so much time from the computer that a delicate timing analysis could be upset. It would be wise to boot your machine from a plain DOS diskette when running these tests for maximum reliability. Also, turn off (or better yet remove) any disk caching software you might be using. Disk caching intercepts the disk reading performed by the programs and renders their measurements invalid (and rather humorous). By the way, disk caching program performance is significantly improved through proper disk interleaving.

## What It All Means

So now you're wondering: What do the numbers mean? How do they compare with industry norms and everyone else's? Should I be dancing on roof tops or wringing my dealer's neck?

A hard disk spins at 3600 revolutions per minute, or 60 revolutions per second. A track has 17 sectors of 512 bytes per sector. This means that data passes under your drive's head at a rate of 522,240 bytes per second. (This number is 768,000 for RLL controllers with 25-sector tracks.) Your system will achieve some fraction of this maximum possible rate as determined by the number of rotations required to read or write each track. The number 522,240 (or 768,000) is divided by the number of revs per track (as SPINTEST does) to calculate your drive's data transfer rate.

Since your system's performance is meaningful only when compared to other properly and improperly interleaved systems, see the table (included with the on-line documentation file INTLEAVE.DOC) to get a feeling for where the industry stands.

We have seen that WD controllers which miss at 3:1 will do beautifully at 4:1. Several InfoWorld readers have independently confirmed that AT&T machines (with WD controllers) perform best at the very loose interleave of 6:1. Even the old true blue IBM controller which runs at 6:1 can always do 5:1 and even 4:1 in a faster than 4.77 MHz machine.

## Determining Factors

So what system factors influence and determine the optimal interleave setting for a given set of equipment? After a sector of data has been read it must be moved from the controller's onboard buffer into the computer. This is done with a process known as DMA (Direct Memory Access). The time required to transfer the sector determines how soon the controller will be ready to read the next sector. For this reason add-in Accelerator "Turbo" Cards do not generally change a machine's optimum interleave since the main system clock speed, which continues to control DMA memory accesses, is not changed. The less expensive "Speed Booster" products which alter the basic system clock timing DO have a tremendous impact on optimal interleave by running the DMA faster and thus moving the data in and out faster.

## Gibson Research and Interleaving

When we began these experiments, we had no idea that so many personal computers were so poorly interleaved. Without the aid of SPINRITE, changing a hard disk's sector interleave manually requires first backing up all hard disk data onto some secure medium. Then a low-level reformatting is performed. This messy task was never designed for the casual computer user since it requires unpublished knowledge of the internal details of your controller, using the DOS DEBUG command to poke hexadecimal values into the 8088's machine registers and starting the low-level format. After this, the FDISK and FORMAT commands are used to create a partition table and lay down the high-level formatting information. After all this, the backed-up data must be copied back onto the drive.

Then, if the experimentally chosen interleave was not correct, most of the process would have to be repeated with a different trial interleave factor. Responding to the clear need for a better solution to the task of hard disk sector interleaving optimization, we developed the now-famous SPINRITE product to automate and streamline this "re-interleaving" process.

SPINRITE quickly determines the optimal interleave setting for any disk of any size in any system with any clock rate. Once determined, the existing interleave of the disk can be RESET automatically and in just a few minutes, leaving all your disk data intact and in place! It also performs several other useful hard disk utility functions which have never before been available.

## Get Your Disk Spinning Right!!

Even if SPINTEST turned in excellent results of 3 or 4 revs, you should still check out Gibson Research's new product for the other surprising (never before offered) hard disk capabilities it brings. If you received results like 10, 12, or even 17, 18, or 19 revolutions, there is no doubt that your system could be running four to five times faster in minutes!

To receive all the details about SPINRITE simply phone or send your name and address either through our 24-hour BBS or through the US Mail. Ask for our completely descriptive SPINRITE literature.

GIBSON RESEARCH CORPORATION
22991 La Cadena
Laguna Hills, CA 92653
Phone: (714) 830-2200
Tech Support: 830-2500
Fax: 830-0300 24-hour
BBS: 830-3300

# STACKEY

**Version 4.0**
**CTRLALT Associates (R)**
**Copyright © 1986–1992 by Barry Simon & Richard M. Wilson**

## Introduction

STACKEY, a utility from the makers of CTRLALT, will automatically place keystrokes in your keyboard buffer. It is intended for use in connection with batch files. Here is a typical example of a batch file "lot.bat" for use with 1-2-3 using STACKEY:

```
stackey W18 CR"/FR%1.wk1" CR
123
```

Entering "lot mysht" at the DOS command line starts the batch file with mysht as %1. STACKEY has the command line passed to it by the batch processor with %1 already replaced by mysht. Thus STACKEY places in the keyboard buffer the text: **/FRmysht.wk1** followed by a carriage return. The initial W18 tells STACKEY to pause about 1 second to allow 1-2-3 to load. If it weren't there 1-2-3 would happily remove the keystrokes from the buffer before allowing any input. These characters wait in the buffer until a program requests keyboard input. The batch file then loads 1-2-3 which requests input and gets it from STACKEY. /FR calls up a dialog to retrieve a worksheet and %1.wk1 followed by a CR loads that worksheet. The CR after the W18 will banish the opening logo screen in those versions with an opening logo.

And this example hardly uses the power of STACKEY: you can stack any function key or legal Alt-key combination; you can even stack today's date or the current directory name. You can stack commands to turn the Num Lock state on and off. You can stack commands to set colors on the CGA and remap colors on the EGA or VGA.

You can stack a PrtSc. You can stack commands to call up SIDEKICK, SIDEKICK PLUS, or any other program that pops up on pairs of shifts. You can invoke CTRLALT PLUS, CAROUSEL, or DESQVIEW. STACKEY has numerous methods for controlling the flow of a STACKEY "script." You can put delays in or have STACKEY delay its playback until a given message appears on the screen or you can have script playback pause until you hit a key. You can pause playback and later restart it with appropriate hotkeys and even flush STACKEY's buffer that way. STACKEY scripts can even branch depending on what hotkey is hit or what message appears on the screen.

Despite its power, ease of use is an important part of STACKEY. Rather than cryptic codes for keys like F1, you need only use F1. There is even a recorder which you can use to make batch file fragments with the keystrokes you want.

STACKEY also has a utilities mode allowing you to use it to replace lots of little utilities that switch monitors or ports, restore or change the cursor, or even reboot your machine. STACKEY comes with a second program BATUTIL, described in a separate manual with which you can manipulate your environment or get input in batch files.

Finally STACKEY has a special mode of interest to programmers giving you access to STACKEY's insides, allowing you to place inline code in the middle of a STACKEY script.

And STACKEY does its magic with only about 1.7K of resident code (with the default buffer size).

For a quick summary of what STACKEY can do, you can call up help any time you are at the DOS command line with the command

```
stackey ?
```

(see the HELP!! Section I.8 below) or you can consult the quick summary at the end of the documentation.

## For the New User

STACKEY is a powerful package which has many features that you won't want to use too often. For that reason the manual is long, but many parts can be skipped when you first start using the program. For the basic stacking commands, you'll want to read the first three sections of Chapter II after reading the next section.

After that, you can be off and running. If you want to use utilities mode, you'll need to read Chapter V; if you want to set colors with STACKEY, you'll need Chapter IV. This latter chapter is long because of the EGA color tutorial which is included. If you have a CGA, you should only read the first four sections. If you've been confused when hearing about int 16H and int 9, you might look at the keyboard basics parts of Chapter VI.

While STACKEY has numerous options to change the way it works, typically set and changed with /../ instructions on the command line, we have set the defaults to be the most reasonable. Thus, you need not worry about these questions unless you find yourself in a situation where you wished STACKEY worked differently in order to properly interface some particular application program. You may be able to tell it to act in an appropriate way.

## New in Version 4.0

Here are the major new features in this version:

- There is a script recorder. This is a separate resident program you only need to load when you want to record a script. A companion program takes the script recorded in memory and writes it to a file either in a format suitable for inclusion in batch files using STACKEY or in a FILEKEY format.
- The string that a scan command searches for can now be enclosed in ".." as an alternate to (..). This allows scans to include the characters ()<> and |.
- There is now a wildcard character for scan string (ASCII 168, the ¿ character).
- The SC and SS scan commands have been supplemented by SJ (Scan Jump) and SU (Scan jump qUietly) commands which will scan for several strings on the screen and jump to a label in the script depending on which string is found.
- The WS wait for a specific key command has been supplemented by a WJ (Wait Jump) command which waits for one of several different keys and jumps to a label in the script depending on which key is hit.
- There is now a resident version of the UNtil command. That is a command which will pause a STACKEY script until a fixed time.
- There is now a flag to track whether $E and $d use the American or European convention on dates.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the Stackey files in the directory \SECRETS\STACKEY. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the Stackey files to a directory that is included in the search path.

**There are eight executable programs**

| | |
|---|---|
| STACKEY.COM | The basic program |
| SKRES.COM | Version for special loading; see Section III.11 |
| EGAPAL.COM | Program to allow STACKEY to control permanent palettes on the EGA |
| VGAPAL.COM | Program to allow STACKEY to control permanent palettes on the VGA |
| BOXES.COMS | mall program for the demonstration batch file CGACOLOR.BAT |
| SKREC.COM | Program to record STACKEY scripts in memory |
| SKGET.COM | Program to take a macro recorded by SKREC and store in a file in STACKEY or FILEKEY format |

**Four demonstration batch files**

| | |
|---|---|
| CGACOLOR.BAT | Demonstrates STACKEY's color commands for the CGA |
| COLOR.BAT | Demonstrates STACKEY's color commands for the EGA by displaying all 64 colors; use a parameter to indicate the time between color changes in half second units (e.g., "color 3" would give 1.5 second change times) |
| SEECOLOR.BAT | Allows you to see any color or colors supported on the EGA by typing in their numbers or color patterns (e.g., "seecolor 14" or "seecolor RrGgB" or "seecolor 23 42") |
| SOUNDS.BAT | Illustrates the {BEEP} command |

**The help files**

| | |
|---|---|
| *STACKEY.HLP | Basic Help file |

***Note from the Author:*** The shareware disks (in *DOS 6 SECRETS*) do not contain all of the files available in the shareware version of Stackey. The omitted files relate to Stackey external interfaces and are mainly of value to programmers. See the Stackey documentation files included on the diskette for source information for the complete set of files].

Because of our desire to keep the basic files on a single disk and to keep download time low, we are encouraging distribution of a 'small' shareware version without the starred files. If you got a version without this file and want it for evaluation, it can be downloaded from our support section of CompuServe as SKREST.EXE (section 12 of PCVENA) or it can be gotten from many disk vendors including PBS (1-800-426-3475). They will send you a disk with the STACKEY and BATUTIL help files for $5 including S&H. If you upload this program to a BBS, please keep STACKEY.HLP as a separate file.

In addition, there are the files associated with BATUTIL discussed in the BATUTIL documentation.

While these programs are the package and may be distributed as such, the original disks from CTRLALT Associates come with the files in compressed format. The file README.COM on the distribution disk will explain how to install the programs. You cannot just use the copy command. If you got them from another source, that source may have used another packing method.

## Loading and Using STACKEY

The first time that STACKEY is invoked, it loads a small resident portion which becomes permanently resident taking about 1.5 kilobytes of RAM for code plus whatever buffer size you choose. By default, the buffer size is 128 'keystrokes' or 256 bytes. This can be changed; see Chapter III — indeed many of the new features takes 3–10 keystrokes of buffer space and some considerably more, so you may want a larger buffer. It displays the message

```
Resident part of STACKEY now in
   place.
```

and proceeds to read its command line. At later times, STACKEY will find itself in memory and only read its command line. If you are using a hardware or software protocol that allows loading in high memory (MAXIT board, 386max, or QEMM), you'll want to use SKRES rather than STACKEY as you will if you want a buffer with over 10,000 keystrokes (see Section III.11).

STACKEY will place keystrokes in its stack according to the rules of syntax described below. If there is a syntax error, it will exit with no keyboard stacking and give an error message. For example, saying

```
stackey "hello "CRthere
```

will result in the error message

```
Oops! There is an error on the
   STACKEY command line.
"hello "CRthere
^
Unrecognized command.
```

The arrow (shown as a caret here) will point to the first place on the line that STACKEY was unable to interpret. The command line to send hello on one line and there on the next should have said

```
stackey "hello "CR"there"
```

The arrow will indicate other errors; for example, if you ask STACKEY to swap printer ports but you only have one printer port, STACKEY will exit with

```
Oops! There is an error on the
  STACKEY command line.
{PRN}
^
Hardware or software environment
  does not appear to support this
  command.
```

The syntax is fine but there is an error none the less. Similarly, if you try to swap Carousel partitions but STACKEY finds that Carousel isn't loaded, you will get an error message.

The command line is parsed into parts consisting of text between quotes, two character special codes, hex and decimal input, delays, and special commands. Some of the two character special commands allow an integer for the second "character" which may be more than one numeral. The special commands include EGA/VGA color settings between square [] brackets and utilities commands inside pointed braces {}. Certain features like SCan and SHift take parameter inside (). If only a utility is called up and the final } is left off, the resident part of STACKEY will not load. Spaces between the parts are optional so that

```
stackey F1"hello"^r@E
```

is the same as

```
stackey F1 "hello" ^r @E
```

or even

```
stackey F1   "hello" ^r@E
```

but

```
stackey F 1"hello"^r@E
```

which splits the two character code. F1 will result in an error message.

## Help!!

Eighteen screens of help are available if you type

```
stackey ?
```

To get help, the file STACKEY.HLP distributed with STACKEY must be available either in the default directory or in your path. This STACKEY.HLP file is different from the files of the same name distributed with STACKEY 2.0 and 3.0. If that old STACKEY.HLP is found, STACKEY will exit with the error message:

```
The file STACKEY.HLP has wrong
  format or version number.
```

The first help screen is a table of contents and the last one an index. You can search by hitting S or goto a given page with Goto. The help is a useful summary but not really a replacement for reading the manual!

On a true monochrome monitor like the original MDA or Hercules card, STACKEY ? will show up in the proper two color attributes. But on a color adapter with monochrome screen (for example, most laptops), the "colors" may make the help invisible or ugly. On such a screen call for help with

```
STACKEY ?m
```

When you exit the help, a screen appears reminding you to register. You can suppress this screen by placing

```
B4$=I paid
```

in your environment. Obviously, having told you the secret, you can do it even if you haven't paid — let your conscience be your guide.

## Errorlevel

If STACKEY can interpret its command line and does not find the buffer full, it exits with an error level of 0. If there is an error in the command line, the ERRORLEVEL is set to 1; if there is a buffer overflow, the ERRORLEVEL is 2 and if there is any kind of DOS file error whether due to an invalid path or an open drive door in the {dumpf} utility, the ERRORLEVEL is set to 3. You can test the ERRORLEVEL in batch files with the **if ERRORLEVEL..** DOS command. All error messages are directed to standard output so you can redirect them to NUL and suppress them if you will handle errors with ERRORLEVEL. An example of this can be found in the demonstration batch file COLOR.BAT. If you enter an illegal parameter value, say **color a**, the batch file responds by suppressing an error message from STACKEY and giving you one appropriate to the batch file. How STACKEY responds to an error is controlled by the /t/ and /c/ commands discussed in Chapter III. By default, STACKEY will flush its and BIOS's buffers when there is an error, but the batch file will continue.

If STACKEY finds that the version number of the resident part does not agree with the version number of the file that you have invoked, it will exit with an ERRORLEVEL of 4.

## Chapter II: Placing Keystrokes in the Buffer

### Stacking Text

Most often, you will want to stack ordinary alphanumeric data, that is the white letters in the center of the keyboard including the numbers, punctuation, braces, etc. You do this by placing the text within quotes on the command line as in

```
stackey "hello there"
```

or

```
stackey 'hello there'
```

Either single or double quotes can be used. Once a quote occurs on the line, every letter counts until a matching quote OF THE SAME TYPE occurs. So to place

```
"Hi!", he said
```

in the stack, you can use

```
stackey '"Hi!", he said'
```

You cannot stack both types of quotes within one string but you can stack strings with both types of quotes by using more than one string. For example, to stack

```
"I can't come", he said
```

use

```
stackey '"I can' "'" 't come", he
  said'
```

Later, we will describe the command codes SQ and DQ for placing quotes. You could also use:

```
stackey DQ"I can't come"DQ", he
  said"
```

to stack the above string.

While you will mainly stack ASCII characters corresponding to the white keys between quotes, you can also stack an ASCII character such as the graphics characters. STACKEY will simulate the hitting of the A key if you use stackey "A"; it will simulate entering an ASCII code above 128 as if it were entered from the Alt-numeric keypad.

### Two-Character Codes

Many keys on the keyboard do not correspond to any ASCII character. For these and also for a few ASCII characters, STACKEY recognizes special two-character codes. The first of these can be a special symbol such as ^, @ or, #. Because some programs like CED give special meaning to some of these codes, we have given them alternatives. That is, as the FIRST character in these codes:

~ means the same as ^
( means the same as @
) means the same as #
* means the same as $

This replacement is only effective for the first character, so that you cannot replace ^^ by ~~ but only by ~^. If you need to stack a control-^ and ^ has a special meaning, you will have to use another method like hex codes. These alternatives are not applicable in the color setting commands (i.e., you cannot use [)3 1=4] in place of [#3 1=4]).

You stack function keys using one of F,S,C,A (for function, shifted function, control function, and alt function) followed by a single digit; 0 stands for 10; on an enhanced keyboard – stands for 11; and = for 12 (note the – and = keys are on the numeric row in the eleventh and twelfth place). Thus Ctrl-F10 is stacked with C0 and

```
stackey C2 A3 S5 F0 A-
```

would stack the sequence Ctrl-F2, Alt-F3, Shift-F5, F10, Alt-F11. You may use either uppercase or lowercase for the key letter f,s,c,a. Do not confuse ^1 (for Control-End) and C1 or @1 (for Alt-1) and A1. You can stack the numbers on the numeric keypad with N0,..., N9, the numeric decimal point (Shift Del) with N, and the grey +, – and * with N+, N– and N*.

The BIOS also makes codes for certain Alt key combinations: legal possibilities are Alt- followed by any letter or number (top row numbers) or the following special symbols: – =. These can be stacked by preceding the key with @, for example

```
stackey @a @2 @-
```

will stack Alt-A followed by Alt-2 followed by Alt-minus. It does not matter if you use upper- or lowercase letters or if you use @! rather than @1 or @_ rather than @–.

Similarly, available control combinations are the letters A-Z, [,\,],^,_,‘,@ left arrow, right arrow, home, end, PgUp, PgDn, PrtSc. These are entered using the ^ (or ~) as the first key. ^A through ^Z will enter those control codes as will ^[, ^\,.... The six keys, on the numeric keypad corresponding to ^Home, etc. can be entered with the corresponding numbers 1,3,4,6,7,9. Thus

```
stackey ^z ^Z ^[ ^6
```

will stack two Ctrl-z's, an escape, and then Ctrl-Right Arrow (but the “escape” here is entered as if you depressed the control key and then struck [; to enter an escape as if you hit the Escape key, use $e or ES as described below). For the letters, ^ may be followed by an upper- or lowercase letter. For the remaining combinations, only one possibility is allowed. For example, do not confuse ^^ and ^6.

In the next chapter, we will discuss the PH two character code which stacks a PHony keystroke and the WR two character code which stacks a “Wait for Request.”

Next, there are two-character mnemonics for special key combinations as follows:

| | |
|---|---|
| LA | Left Arrow |
| RA | Right Arrow |
| UA | Up Arrow |
| DA | Down Arrow |
| PU | Page Up |
| PD | Page Down |
| HM | Home |
| EN | End |
| IN | Insert |
| DE | Delete |
| G+ | Grey plus (“+” will send the top row +; also N+) |
| G– | Grey minus (“–” will send the top row –; also N–) |
| G* | Grey star (“*” will send the top row *; also N*) |
| CP | Control PrtSc |
| PS | PrtSc |
| TA or TB | Tab |
| ST or BT | Shift Tab (=Back Tab) |
| ES | Escape (also $e) |
| BS | Backspace (also $h) |
| SP | Spacebar (also “ ”) |
| CR | Enter (carriage return) |
| LF | Ctrl-Enter (line feed) |
| FF | Form Feed (^L) |
| DQ | The double quote “ |
| SQ | The single quote ‘ |
| CB | Ctrl-Backspace |

These codes can be entered in upper- or lowercase. It is a little known fact that Ctrl-Backspace is a legitimate keystroke that even enters a valid code: ASCII 127.

PS for PrtSc will simulate the pressing of the Shift-PrtSc key. If some other program, for example PIZAZZ PLUS, has taken over that key, it will be called up with PS.

^! will simulate the Ctrl-Break key; for techies, it places an ASCII 0 word in the keyboard buffer and then calls int 1BH.

STACKEY provides support for combinations understood by the enhanced keyboard or by the BIOS that comes with machines supporting the enhanced keyboard. First, the following Alt-key combinations are allowed: Alt-[ Alt-] Alt-; Alt-’ Alt-‘ Alt-\ Alt-, Alt-. Alt-/. These are entered with the combinations A[ A] A; A’ A‘ A\ A, A. A/. In addition we have the following codes for the enhanced keyboard:

| | |
|---|---|
| Alt-Bksp | AB |
| Alt-Enter | AC |
| Alt-Esc | AE |
| Alt-Tab | AT |
| Ctrl-Tab | CT |

The Alt-keypad cursor keys (i.e., everything but the 5 key but including Ins and Del) are allowed keys with an enhanced keyboard. You enter them with K followed by the number, (e.g., Alt-Left is K4 and Alt-Del is K, and the center key [unshifted 5 when numlock isn't on] is K5). In addition we have

| | |
|---|---|
| Alt-Numpad K / | K/ |
| Alt-Numpad K * | K* |
| Alt-Numpad K – | K– |
| Alt-Numpad K + | K+ |
| Alt-Numpad K Enter | KC |
| Ctrl- K / | ^/ |
| Ctrl- K * | ^* |
| Ctrl- K – | ^– |
| Ctrl- K + | ^+ |
| Ctrl- K Up [8] | ^8 |
| Ctrl- K 5 [5] | ^5 |
| Ctrl- K Dn [2] | ^2 |
| Ctrl- K Ins[0] | ^0 |
| Ctrl- K Del[.] | ^. |
| Grey Enter | GE or GC |

While most programs do not distinguish them, the enhanced BIOS will distinguish the cursor pad keys and the numeric pad keys. The cursor pad keys are given by

| | |
|---|---|
| Cursor pad End | G1 |
| Cursor pad DA | G2 |
| Cursor pad PD | G3 |
| Cursor pad LA | G4 |
| Cursor pad RA | G6 |
| Cursor pad Hm | G7 |
| Cursor pad UA | G8 |
| Cursor pad PU | G9 |
| Cursor pad Ins | G0 |
| Cursor pad Del | G. |

The logic behind the codes is a little stretched. The G is because the keys are grey; the 1, 2, 3 correspond to the numbers the keys correspond to on the keypad (not the cursor pad!).

An appendix to this chapter summarizes all the two-character codes in alphabetic order.

## Status Toggles

You can turn NumLock, etc., on or off from the STACKEY command line. The syntax is

| | |
|---|---|
| +N | Turn NumLock on |
| –N | Turn NumLock off |
| +C | Turn CapsLock on |
| –C | Turn CapsLock off |
| +S | Turn ScrollLock on |
| –S | Turn ScrollLock off |
| +I | Turn BIOS insert mode indicator on |
| –I | Turn BIOS insert mode indicator off |

Strictly speaking, the discussion of status toggles does not belong in this chapter since they are handled in real time. That is, rather than placing a command in the resident buffer, a command like +N is interpreted and acted on by the non-resident part of STACKEY as soon as it reaches the interpretation of that part of the command line. In this sense, the lock toggles are like utilities and, indeed, they are included also in utilities mode. They continue to be available in two-code form for compatibility with earlier versions of STACKEY. BATUTIL will let you read the state of the lock keys from a batch file; see Section IV.9 of the BATUTIL documentation.

## Prompt Metastrings

STACKEY supports the full set of metastrings allowed in the DOS PROMPT command as well as the added strings supported by the program SEND.COM of Howard Rumsey and Barry Simon. Specifically:

| | |
|---|---|
| $$ | The character "$" |
| $t | The time in HH:MM:SS.hh format |
| $d | The date in DAY MM-DD-YYYY format (e.g., Tue 9-30-1986) |
| $p | The current path in full (e.g., C:\BIN\FOO) |
| $v | The current DOS version |

| | |
|---|---|
| $n | The current default drive |
| $g | The character ">" |
| $l | The character "<" |
| $b | The character "\|" |
| $q | The character "=" |
| $h | The backspace |
| $e | The ESCape |
| $_ | CR/LF (i.e., <Enter> followed by Ctrl-<Enter>) |
| $P | Same as $p in the root dir and as $p\ elsewhere |
| $T | Time in HHMM format |
| $M | Month in MM format (e.g., 09) |
| $D | Day in DD format (e.g., 03) |
| $Y | Year in YY format (e.g., 86 in 1986 and 01 in 2001) |
| $W | Day of the week in English (e.g., Sunday) |
| $E | The date in English (e.g., February 18, 1988) |
| $H | Hour from 00 to 23 |
| $m | Minute from 00 to 59 |

There is a difference between uppercase and lowercase after the $. Because of STACKEY's send mode (see Section II.10), you need no longer use the SEND program but can use STACKEY instead. That is, you can use STACKEY to send these codes to standard output.

STACKEY keeps an internal flag that affects the meaning of $E. When the flag is American (the default but it can be set with /a/), $E will be, for example, February 18, 1988 and $d is, for example, Thu 2-18-1988. When the flag is European (set with /e/), $E would be, for example, 18 February 1988 and $d Thu 18/2/1988. If you want the European effect all the time, load STACKEY with

```
stackey /e/
```

## Decimal and Hex Codes

This section deals with advanced features not needed by most users. See Chapter VI for a tutorial on keyboard basics used in this section.

You may enter various ASCII key combinations from the keyboard by pressing Alt (or by pressing Alt-Left Shift) and tapping one to three digits on the keypad and then releasing Alt. To stack keystrokes as if entered in this way, use \ followed by a decimal number, so

```
stackey \104\101\108\108\111
```

at the DOS command line will cause **hello** to appear on your screen.

Keystrokes are stored in the buffer as a set of two hex bytes: a scan code and an ASCII code. You can enter such a combination by typing an X followed by four hex digits. The first two are the scan code and the second pair the ASCII code. For example

```
stackey X2368 X1265 X266C X266C
    X186F
```

at the DOS command line will cause **hello** to appear on your screen. The hex digits a–f and the letter x can be either upper- or lowercase.

Because they are used internally by STACKEY, the scan codes E0 through FF (which are not legal anyway) are allowed but take the place of two keystrokes in STACKEY's buffer. STACKEY will place in the keyboard buffer whatever combination you tell it to, even if it is an illegal combination. DOS will only pay attention to the ASCII half of such combination as will many programs. Some keyboard macro programs will allow such possibilities. For example, KEYWORKS associates Ctrl-Up Arrow (a keystroke not supported by an old style BIOS) with 8448. Stackey X8448 will call up the KEYWORKS macro assigned to Ctrl-Up if one is defined. Some combinations are filtered out by the so-called enhanced keyboard.

## Invoking Special Programs and Keystrokes

Most TSRs will not pop up by responding to strokes in the keyboard buffer and, as a result, STACKEY cannot invoke them by its usual method. However, when we have special information about a program, we can provide a special routine for that program in particular. We'll discuss those commands, all new in Version 3.0 here. For programs that pop up on depression of several shift keys, like SIDEKICK, we can simulate depressing shift keys as discussed in the next section. Here are the special invocation routines:

CTRLALT PLUS: CA will invoke the main menu of CTRLALT PLUS so, for example

```
stackey CA"MAP"
```

will print the entire screen even if you are in 43 or 50 line mode. If CTRLALT PLUS is not loaded, you'll get an error message from STACKEY.

DESQVIEW: Quarterdeck provides two methods for macros to simulate the Desq key which work on different keyboards or BIOS. We have assigned them to D1 and D2. So for example

```
stackey D1"Z"
```

would zoom the current window to full screen. The combination DL is the same as the Desqview learn key. Finally {boot=d} as a utility will invoke a Desqview specific reboot command.

CAROUSEL: You can tell Carousel to switch to partition n with Qn where n=1,2,...,9,0 (0 is for partition 10); with Carousel 3.0 or later, you can also use Q– and Q= to switch to partitions 11 and 12. Thus you might have a sequence of batch files that successively loads partitions 1,2,.... If you want to load program "foobar" in partition 6, issue the string "hello" followed by F1 to the program and then switch to partition 7, you'd end the batch file for partition 6 with

```
stackey "hello" F1 W36 Q7
foobar
```

The wait command W36 which is discussed in the next chapter (and pauses for 2 seconds) gives the program time to react to the keystrokes before the swap. You may need to adjust the W command for particular applications.

TESSERACT PROGRAMS: Tesseract is a library of TSR routines provided by Innovative Data Concepts as a shareware product. If a TSR is written with these routines, you can invoke it with the TE command:

```
stackey TE(idstring)
```

will call up the tesseract TSR with the given idstring name. You'll need to consult the documentation that came with your program to find the required idstring, but here is an example. Suppose you have SWAPSP and want to dial the third entry in your phone book under M. Use

```
stackey TE(SWAPSP15)"PM"DNDNCR
```

The idstring must be exactly eight characters in length. For Tesseract programs that allow multiple hotkeys, each hotkey has a number. TE(idstring) will call the program with hotkey 0. For other hotkeys, place a /NN after the idstring. Thus, for example

```
stackey TE(idstring/23)
```

would invoke the program in question via hotkey 23.

CLONE SPEEDUP/SLOWDOWN: Many clones have two speeds adjusted with ^@G+/–. You can simulate this with

```
stackey Z+
```

to speedup and

```
stackey Z-
```

for slowdown. This will not work with all clones.

You might object to some of these additions to STACKEY by saying that you don't use CAROUSEL and object to resident memory being taken by the CAROUSEL command. In fact, most of these commands take NO additional memory — they use the inline feature discussed in Section VI.7.

## Simulating Depression of Shift Keys

Some popups come up when you depress a pair of shift keys. You can simulate the depression of any subset of the four shifts with the command SH which takes two parameters in the form

```
SH(XXX,YY)
```

where XXX is some subset of the letter C,A,L,R for the Control, Alt, Left shift, and Right shift respectively and YY is a number from 1 to 65,535. This command simulates depressing the shift keys in question for YY clock ticks. You'll need to experiment with the value of YY depending on the program. We added that parameter because different programs seem to require different values. For example, with a default SIDEKICK Ctrl+Alt, you can invoke the SIDEKICK main menu from within a STACKEY script with

```
stackey SH(CA,1)
```

but invocation of SIDEKICK PLUS requires

```
stackey SH(CA,n)
```

where n is at least 7 or 8 while PRIMETIME (with say C+L as hotkey) seems to require

```
stackey SH(CL,10)
```

You'll need to adjust the shift mask to whatever combination you've chosen for your application. For example, if you've changed the SIDEKICK popup to C+L, use SH(CL,1) rather than SH(CA,1).

## Stacking Real Time Beeps

STACKEY scripts can include real time BIOS beeps with the special two key code BE. Beeps are especially useful combined with loops and waits and with jumps (see Section III.2). II.9

## Pop-up Messages

STACKEY scripts can include pop-up messages that are displayed and take over the machine until the user hits a key. There are two versions depending on whether you want the key hit passed to the underlying application.

```
STACKEY MP(hi there)
```

will display the message **hi there** in a box and pass the keystroke on while

```
STACKEY ME(hi there)
```

would eat the keystroke. These messages are implemented as inline code to avoid having to permanently use resident memory but as a result they take a lot of buffer space. A message takes 156 bytes plus 7 times the length of the message. The default buffer has only 256 bytes so that long messages will require you to load STACKEY with the command (see Sections III.11 and III.12) to take a larger buffer.

## Sending to Standard Output

STACKEY places keystrokes in the keyboard buffer. However, you can tell STACKEY to instead send the interpreted material to standard output. You could then redirect this output to a file or the printer. Thus STACKEY provides a replacement for the SEND program of Rumsey and Simon. With their permission, a modified version of the documentation for SEND is provided as an appendix to this chapter.

To send output to the standard output, include the special symbol underscore in the STACKEY command line so that STACKEY _"hello"$_ will have the same effect as

```
echo hello
```

Following the character _ (underscore) the ONLY allowed codes are

- character strings in quotes
- control codes of the form ^x with x=a,...,z,[,\,],^ or _ (but not ^1, or ^@ etc)
- prompt metastrings in extended form as discussed above
- upper ASCII codes in the form \xxx
- the following two character codes: TA, TB, ES, BS, SP, CR, LF, FF, SQ, BQ, CB

Any other codes, even ones normally legitimate in STACKEY (e.g., LA which doesn't have an ASCII equivalent) will produce an error message but only AFTER the codes prior to it on the command line are already sent to standard output.

The special command _ is only in effect for the line it is issued on; you must include it on each line which you wish to send to standard output. It need not be the first command on the line and a second _ will toggle back to real mode.

If you have CED or a similar command-line processor that allows synonyms or aliases, we suggest that you use CED SYN send 'stackey _&a'

# Appendix to Chapter VIII

Here is a quick summary of the STACKEY command-line syntax:

- ? will give help
- _ will send the remainder of the line to standard output
- Place text inside single or double quotes
- F1-F0,F–,F=; S1-S0,S–,S=; A1-A0,A–,A=; C1-C0,C–,C= for function keys
- @x or (x for Alt-x where x=letter or number
- ^x or ~x for Ctrl-x where x=letter or number

(***Note:*** @number for top row numbers; ^number for keypad)

- LA,RA,UA,DA for arrow keys
- PU,PD,HM,EN for Page Up/Dn, Home, End
- IN,DE for Insert/Delete
- N0,...,N9,N. for numeric keypad numbers and decimal point
- G+,G–,G* or N+,N–,N* for Grey keys
- PS for PrtSc
- CP for Ctrl-PrtSc, CB for Ctrl-Backspace
- TA or TB for Tab; ST or BT for Shift Tab
- ES,BS,SP,CR for Escape, Backspace, Space, Enter
- FF, LF for form feed (^L) and Ctrl-Enter
- DQ,SQ for “ and ‘
- PH and WR for two kinds of “phony keystroke”
- +/–N,+/–S,+/–C,+/–I to turn On/Off Num, Scroll, Caps Lock, Insert
- SH(X,nnn) with X a subset of CALR simulates shifts for nnn ticks
- CA invokes CTRLALT PLUS
- Qn switches to Carousel partition n
- Dn with n=1 or 2 invokes Desqview main menu
- TE(idstring,keyid) for Tesseract programs
- Z+/– clone speedup/slowdown
- P–/+ to disable, reenable Shift-PrtScrn
- BE for BIOS beeps
- MP(msg), ME(msg) for pop-up messages
- $x or *x for prompt metastrings with SEND extensions (see following)
- \nnn with nnn=decimal number for Alt-keypad entry
- Xhhhh with hhhh=hex word for direct hex entry
- #nnn or )nnn to repeat the next keystroke nnn times
- :nnn label (nnn=0 to 255)
- Jnnn jump to label nnnn
- Wnnn to insert a delay of nnn ticks (18.2 ticks = 1 sec)
- Mnnn to insert a delay of nnn minutes

- Jnnn jump to label nnnn
- Wnnn to insert a delay of nnn ticks (18.2 ticks = 1 sec)
- Mnnn to insert a delay of nnn minutes

- UNhh:mm resident until command
- SC(string) to scan at cursor position
- SCn,m(string) to scan at absolute position
- SS(string) and SSn,m(string) to scan silently
- SJ(string1; string2;....; string n) to scan multiple strings
- SU(string1; string2;....; string n) to scan multiple strings silently
- WB pause script until buffer is empty
- WH pause script for restart hotkey
- WK pause script for any key Convention: X=subset of CALR; Y=single keystroke like “a” or F1
- WS(X,Y) pause for specific key
- WJ(X1,Y1;X2,Y2;...Xn,Yn) pause for multiple hotkeys
- H!(X,Y) resets flush buffer hotkey (default=AL,!)
- H*(X,Y) resets flush scan hotkey (default=AL,G*)
- H–(X,Y) resets stop hotkey (default=AL, G–)
- H+(X,Y) resets restart hotkey (default=AL,G+)
- CL,NL,SL,SR CapsL, NumL, ScrollL, SysReq for hotkey only
- ! to flush buffer

- ; to stop reading command line (for comments)
- /Bnnnn/ to change buffer size on initial loading
- /Rnnnn/ to force extra loading with new buffer size
- /i/ /s/ and /p/ for impatient, patient, smart processing
- /t/ and /n/ to turn test mode on and off
- /c/ and /d/ to turn continue mode on and off
- /u/ and /o/ to push and pop the s,t,c flags
- /a/ and /e/ to set American vs European dates for $E & $d
- /?/ to query the s,t,c flags
- Ln to set coLors on the CGA (n=0,...,15)
- Pn to set palette on the CGA in mode 4 (n=0,1)
- [c=nn] to map color number nn to slot c on the EGA
- [#m;c=nn] to do the map in the permanent table for mode m
  - #A:All palettes
  - #T:Ultravision, text modes only
  - #U:Ultravision
- +/–B to turn blinking on/off on the EGA or VGA
- IL(xx/xx/xx/xx) InLine command (USE WITH CAUTION!)
- EX(filename) External inline (USE WITH CAUTION!)

In summarizing utilities mode, minimal truncations are capitalized. Only use the leading capitol letter for x.

- {BOOT} or {REBOOT} for a warm reboot
- {BOOT=C} or {REBOOT=C} for a cold reboot
- {MONitor=x} x=Color,Mono,Switch
- {Prn} to switch LPT1 and LPT2
- {COmm} to switch COM1 and COM2
- {CUrsor=x} x=+/– to turn standard cursor on/off
- {CUrsor=n,m} to put cursor on scan lines n through m
- {CPosition=n,m} to move the cursor to row n, column m
- {MODe=n} switch to allowed video mode
- {LInes=n} n=25,43(50) to shift modes on the EGA (VGA)
- {LO=xy} where x is + or – and y=N,S,C or I to toggle lock keys
- {Wait=nnn} Real time pause of nnn clock ticks; n between 1 and 32K
- {Until=HH:MM} Real time pause until time HH:MM
- {DUMPP} to do a screen dump to printer
- {DUMPF=x}, {DUMPL=x} to append screen to file with filespec "x"
- {DUMPOF=x}, {DUMPOL=x} to overwrite screen to file
- {HALT} stops batch file with ^C
- {Keywait}, {Enterwait} pause for input
- {Abortwait}, {~abortwait} pause for input to abort
- {BEep=nn,mm} nn = inverse pitch, mm = duration PROMPT metastrings and extensions:
- $v = DOS version
- $n = current drive
- $p = current path
- $P = current path with \
- $$ = the character $
- $g = the character >
- $l = the character <
- $q = the character =
- $h = the character bks
- $b = the character |
- $e = the character esc
- $_ = CR/LF-
- $t = time in HH:MM:SS.hh format
- $T = time in HHMM format
- $H = hour from 0 to 23
- $d = date in DAY MM-DD-YYYY
- $M = month in MM format
- $D = day in DD format
- $Y = year in YY format
- $W = day of the week
- $E = date in English

FILEKEY has the following conventions:

- default interprets as pure text
- $$ toggles between pure text and STACKEY command mode
- $; in text mode suppresses CR and remainder of line

## STACKEY & BATUTIL Registration Form

Name: ______________________

Company: ______________________

Address: ______________________

City: ______________________

State: ________ Country: ______________

Zip: ________ Phone: ______________

**Remit to:**

Advanced Support Group
11900 Grant Place
Des Peres, MO 63131
1(800)872-4768
1(314)965-5630
FAX 1(314) 966-1833

You may register by phoning or writing (9 a.m.-5 p.m. Eastern time).

Registration fees are as follows:

License to STACKEY & BATUTIL with printed docs $49

Upgrade to version 3.0 registered users $25

All registrations will get you the printed documentation and the latest version on disk. The printed documentation is essentially identical to the documentation on disk. In addition, there is a $4 shipping and handling fee for registrations in the US and Canada and $13 outside the US/Canada (the package is sent via Air Freight). These are the fees for our US registration point. Please see on-line documentation for additional details on licensing.

Visa, Mastercard, and American Express are accepted. Registrations may also be sent to Advanced Support Group via their CompuServe mailbox Advanced Support Group 70304,3642.

Registration gives you the right to use any version of STACKEY and BATUTIL with a major version number of 4. If there is a version 5, an update fee may be required.

Registered users may obtain support in several ways. Please see on-line documentation for details about support and about multiple-copy discounts. BATUTIL and STACKEY are not normally sold unbundled but only together. Some exceptions apply.

# UMB_DRVR.SYS & UMB_EMS4.SYS Device Drivers

UMB_DRVR.SYS and UMB_EMS4.SYS are device drivers for memory management on DOS-based systems. Please carefully read the documentation for each of these programs prior to installation. It is important to have thorough understanding of memory management before making changes on your system.

## UMB_DRVR.SYS Device Driver

**UMB provider for 286/386/486 systems**
**Version 5.22**
**Copyright © 1991, 1992 by Christopher Blum**

### Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the UMB files in the directory \SECRETS\UMBDRV. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the UMB files to a directory that is included in the search path.

### Introduction

UMB_DRVR.SYS is a device driver that will use the *shadow RAM* capability of the memory controller portion of many chip sets to A) force all ROMs not specifically excluded to be shadowed, and B) expand DOS base memory beyond 640K if possible, and C) provide UMBs (Upper Memory Blocks) to DOS for loading programs and device drivers into upper memory while *not* using any memory below 640K and remaining in *real* mode.

One advantage of this driver is that many if not all other device drivers and TSR programs may be loaded *high* including HIMEM.SYS (even though the DOS documentation says not!).

A second advantage of using UMB_DRVR is that some device drivers that cannot be loaded high when a software EMS emulator is providing the Upper Memory Area because of their use of DMA I/O (this includes some CD-ROM drivers, for example) will work with UMB_DRVR.SYS. This has to do with the characteristics of virtual-86 mode, 386+ memory management facilities, and DMA hardware interactions. See MISCELLANEOUS NOTES — DMA ACCESS TO UPPER MEMORY for more information and possible restrictions.

In addition, remaining in real mode allows programs that must be able to use protected or virtual-86 mode themselves, such as Borland's Turbo Debugger (TD386.EXE / TDH386.SYS), to operate as intended (and yes, TDH386.SYS can be loaded high with no problems).

## Definitions

Hex addresses are given in full hex notation as opposed to Intel segment:offset form (i.e., A0000 in full hex is the same as A000:0 in seg:off form). The memory sizes are referred to in Kilobytes (1,024 decimal), Megabytes (1,048,576 decimal), and Gigabytes (1,073,741,824 decimal).

BASE MEMORY — RAM available to DOS and programs from location 0 to 640K-1 (9FFFF hex). All programs have access to this area.

UPPER MEMORY — The area between 640K (A0000) and 1M-1 (FFFFF). This is the area where ROMs on expansion cards reside (usually), where the EMS base area (the *window* into EMS memory) is (usually), and where Upper Memory Blocks (UMBs) are created for loading device drivers, programs, etc., high with the DOS 5 DEVICEHIGH and LOADHIGH commands. DOS does not create UMBs itself, but rather relies on a program called a *UMB provider* to supply them. DOS then manages the upper memory area as an extension of base memory with special characteristics when you use the DOS=xxxx,UMB command. Programs like UMB_DRVR, QEMM, 386^MAX, and others are UMB providers.

HIGH MEMORY AREA (HMA) — The HMA is memory from 1M (100000) to 1M+64K-16 (10FFF0) (i.e., the first 64K-16 bytes of extended memory). It can be accessed on 286 and up CPUs in real mode because the address calculation logic does not wrap to location 0 from FFFFF, allowing a program to use the segment *FFFF* to access memory up to 10FFF0. On the earlier 8088 and 8086 processors, the wrap to location 0 was used by some software. To maintain compatibility, system designers have included a way to make the newer CPUs act like the older ones. It is a *gate* that can allow the CPU's address line 20 (A20) to be held to 0 (emulating the behavior of the older CPUs), or to be passed through. With DOS 5, use of the HMA must be through a program which controls access to it by opening this gate for times the HMA must be accessed and closing it so that other programs cannot accidently get at the HMA. The DOS 5 program which performs this function is the device driver *HIMEM.SYS*. Other programs such as QEMM, 386^MAX et.al. also provide this function. The HMA is managed as a total unit (i.e., only one program "owns" it, and "owns" it all). This is where most of DOS is placed when you use HIMEM.SYS and the DOS=HIGH command. As stated before, when DOS is loaded high, no other program can use this portion of memory. The definition of this area and its use is standardized in the Extended Memory Specification (XMS) issued by Microsoft / Lotus / Intel / AST Research, although there is question as to where the credit for the discovery of the area and its first usage in real mode should go.

EXTENDED MEMORY — Memory starting at 1M (100000) that is accessible by the CPU in protected mode. On a 286, this range extends up to 16M-1 (FFFFFF), and on 386 and above CPUs up to 16M-1 (FFFFFF) in 16-bit mode and up to 4G-1 (FFFFFFFF) in 32-bit mode (not used by vanilla DOS, but possibly by some DOS extenders). Under DOS, this memory is accessed in several ways:

1. BIOS INT 15H functions — This method is the oldest and least standardized, but requires no special drivers. Programs directly access the BIOS functions to utilize the memory, and must take great pains to avoid "stepping on" other users — many different methods of *marking* used memory exist, even not marking at all.
2. DOS EXTENDERS — These facilities are supplied by several vendors including Phar-Lap and others. They are included within a program and allow that program access to extended memory using the extender's own techniques (usually in protected mode).
3. XMS functions — This method is defined in the XMS standard that was mentioned earlier. It offers a way for many different programs to concurrently use extended memory easily without worrying about the underlying memory management problems. This method is the one used by all of the DOS 5 utilities that use extended memory to provide their services. A device driver is required to provide the XMS services. The DOS 5 driver is HIMEM.SYS. QEMM and other programs also provide XMS access. Most XMS servers including HIMEM.SYS will allow some portion of extended memory to be left outside their control so that programs using the BIOS INT 15H method can still work. Note that DOS 5 *requires* XMS services to access the HMA to load the major part of itself high.

EXPANDED MEMORY — This is memory that conforms to the Expanded Memory Specification put out by Lotus / Intel / Microsoft. It is sometimes referred to as EMS, LIM 3.2, or LIM 4.0 memory. This type of memory is not directly addressable by the CPU, but requires use of additional facilities to be accessed. This memory is available via multiple 16K *pages* in a (usually) 64K window called the EMS base page area within the Upper Memory Area, starting somewhere between C0000 and E0000 on a 16K boundary. The CPU can access this window in real mode and uses the support facilities to map different pages into the window. Although the CPU can only access EMS memory totalling the window size at any one time, it can *move* the window to access all of the expanded memory available.

Expanded (EMS) memory can be implemented in several ways:

1. HARDWARE — Hardware support outside the CPU (usually within the support chip set on the motherboard, or on an expansion slot card) handles the mapping of the memory, controlled by a software driver which merely flips hardware *switches*, and the system runs under DOS in real mode with very little EMS management CPU overhead.
2. SIMULATED — This approach uses extended memory to simulate expanded memory by moving 16K pages back and forth between extended memory and the window (usually *below* 640K, which reduces the base memory area by the window size). Although it has the disadvantages of (usually) reducing base memory and increasing the EMS management CPU overhead, it runs in real mode on any 286 or higher processor without requiring anything more than a software driver. This is the only software option available for many 286 systems. An example of this type of driver is UMB_EMS4, distributed with this package (which, by the way, does *not* reduce your DOS base memory because it uses a 64K window in the upper memory area!).
3. EMULATED — This technique is a sort of cross between 1 and 2. It uses the paging hardware built into 386 and newer processors in conjunction with virtual-86 mode to do the mapping tasks required to provide EMS memory. It is similar to 1 in that the mapping is really done by the hardware, and to 2 in that it also involves nontrivial software to provide the virtual-86 mode environment necessary for it to work. Its advantages are that it works on any 386 or newer CPU without any other special hardware and does not reduce the base memory like 2, but it also has the drawback of restrictions, additional overhead, and complexity introduced by virtual-86 mode. There are several packages that support this type of EMS, including EMM386.EXE supplied with DOS

5, and programs like QEMM, 386^MAX, NETROOM, and Memory Commander. Additionally, these EMS emulators can provide the Upper Memory Area using the same techniques, and generally are good at providing enhanced system functionality for a relatively modest impact on system processor overhead if you have the proper CPU.

With respect to performance of UMB_DRVR.SYS versus the software EMS emulator EMM386.EXE supplied with DOS 5, here are some benchmark results supplied by a (happy) user:

> "System: 386SX 20MHz, VLSI TOPCAT chip set, 4MB RAM, no math processor. DOS version: MS/DOS 5.0 UMB_DRVR.SYS version: 5.09 Benchmark: CHECKIT 3.0 main system benchmark.
>
> | *CONFIG.SYS* | *Dhrystones* | *Whetstones* |
> |---|---|---|
> | None | 3767 | 76.7K |
> | `DEVICE=C:\UMB_DRVR.SYS /C=13`<br>`DEVICEHIGH=C:\DOS\HIMEM.SYS`<br>`DEVICEHIGH=C:\DOS\ANSI.SYS` | 4042 | 77.1K |
> | `DEVICE=C:\DOS\HIMEM.SYS`<br>`DEVICE=C:\DOS\EMM386.SYS NOEMS`<br>`DEVICEHIGH=C:\DOS\ANSI.SYS` | 3683 | 45.7K |
>
> As you can see, there is a significant difference when using UMB_DRVR, not to mention the extra memory saved below 640K. The benchmarks ran faster with UMB_DRVR than they did with no CONFIG.SYS at all. "

***Important:*** The driver must be installed *before* HIMEM.SYS is installed.

It is an XMS 2.0 server providing UMBs to DOS via that standard. The chip set parameter is processed and the proper routine called to remap the unused shadow RAM to DOS-usable memory. Available memory starting at A0000 is used to expand DOS base memory beyond 640K, and other available memory (i.e., above the video memory) is used for UMBs (the areas DOS uses for DEVICEHIGH or LOADHIGH commands).

The driver by default will not use any areas used for video memory. It also forces all ROMs including the system BIOS (F0000-FFFFF) to be shadowed unless forced to be excluded (refer to MISCELLANEOUS NOTES for considerations concerning disk controllers and network cards). If the BIOS has a *boot page* at F0000-F7FFF that the driver can recognize (containing system/CMOS setup code — AMI is one brand that has this), that area will be mapped in as available upper memory since it is not needed after boot time.

The driver should be loaded as the first driver if possible. This allows following drivers and resident programs to be loaded high — even HIMEM.SYS and EMM386.SYS (DOS documentation says they can't, but it works; see MISCELLANEOUS NOTES — WINDOWS and EMS DRIVERS, however). It will initialize, supply UMBs, and terminate leaving a small stub above 640K. To ensure proper chip set function, all warm boots (CTL+ALT+DEL) will be forced to be cold boots after UMB_DRVR is installed. Appropriate status and error messages are issued during processing and a map of the driver's action is displayed.

One of my systems is a 386SX with the Chips and Technologies NEATsx chip set and an AMI (American Megatrend) BIOS dated 04/09/90. It has 4MB of RAM and a Hercules Monochrome Graphics card. I load DOS into the HMA using HIMEM.SYS, supply simulated EMS from the XMS memory pool using UMB_EMS4, and load Borland's Turbo Debugger device driver TDH386.SYS for 386 virtual debugging (TD386.EXE). Using UMB_DRVR defaults and loading all drivers (except SETVER) high gives me 704K base memory for DOS, a maximum executable program size of almost 689K, and 224K in one UMB located at C0000-F7FFF with over 143K still free in that upper memory block for any other TSRs or drivers I may want to load. The following information is extracted from that system (Note: 1K = 1024 decimal).

UMB_DRVR.SYS initializes showing:

```
Chip-controlled RAM at:
                 AAAABBBBCCCCDDDDEEEEFFFF
                 048C048C048C048C048C048C
has been
configured
as:              DDDDvvvvUUUUUUUUUUUUUUss
DOS base memory expansion = 64K
Upper memory block (UMB) area = 224K
[D]OS base memory,
[e]ms base page area,
[s]hadowed ROM,
[U]pper memory area,
[v]ideo, [-]excluded
```

The command **MEM /C** (in DOS 5) displays the following:

```
Conventional Memory:
               Size in              Size in
Name           Decimal              Hex
MSDOS          12304    (12.0K)     3010
SETVER         400      (0.4K)      190
COMMAND        2624     (2.6K)      A40
FREE           64       (0.1K)      40
FREE           705328   (688.8K)    AC330
Total FREE:    705392   (688.9K)

Upper Memory:
               Size in              Size in
Name           Decimal              Hex
SYSTEM         65712    (64.2K)     100B0
HIMEM          1072     (1.0K)      430
UMB_EMS4       73136    (71.4K)     11DB0
TDH386         7920     (7.7K)      1EF0
FREE           146976   (143.5K)    23E20
Total FREE:    146976   (143.5K)

Total bytes available to programs
(Conventional+Upper):
                            852368 (832.4K)
Largest executable
program size:               705184 (688.7K)
Largest available
upper memory block:         146976 (143.5K)
   3080192 bytes total EMS memory
   3080192 bytes free EMS memory

3145728 bytes total contiguous
extended memory
   0 bytes available contiguous
   extended memory
3080192 bytes available XMS memory
   MS-DOS resident in High Memory
   Area
```

***Note:*** If the video card were a CGA, the DOS base RAM expansion would be 96K(!) with executable program size a whopping 721K(!) — a pretty good cure for *RAM cram*! These same results can (and have been repeatedly) achieved on 286 machines!

## Miscellaneous notes

### Boot page

If UMB_DRVR.SYS uses the boot page area (see INTRODUCTION for definition) by default and your system crashes, you need to use the /M= parameter to force it to be part of the BIOS (use ## or —). If no boot page is recognized, you may still try the /M= parameter (with ++ for F0000-F7FFF) if you are brave enough. Heed the warning about having a bootable diskette, however — you may need it.

EMS DRIVERS (EMM386, QEMM, 386^MAX, NETROOM, MEMORY COMMANDER, ETC)

If you run EMS, it is most efficient in terms of contiguous memory to have your EMS base address immediately following your video RAM and any adjacent ROM (e.g., C0000-CFFFF for CGA or monochrome, C8000-D7FFF for VGA) or at the top of the usable area (e.g., E8000-F7FFF with a boot page, E0000-EFFFF without). It is also more efficient in terms of CPU usage overhead to use the hardware EMS driver for your chip set or your EMS memory card instead of a software emulation; see "Introduction" (virtual-86 mode, benchmark).

If you use an EMS driver (hardware or software emulation), you should use the /M= parameter to force UMB_DRVR to exclude the EMS base area. Make sure you *do not* have your driver try to map in the upper memory (640K – 1M) area (other than the EMS base area) — UMB_DRVR has done that already (refer to your driver's documentation). You should be able to use DEVICEHIGH/LOADHIGH to put your driver into upper memory in most cases. If you do not have an EMS driver, try UMB_EMS4 (please refer to UMB_EMS4.DOC). There have been some problems reported running EMM386 with UMB_DRVR — I recommend against it. UMB_EMS4 should provide the services you need.

### ROMS that cannot be shadowed (disk controllers, network cards)

Some ROMs cannot be shadowed by normal means because they have some RAM or a memory-mapped I/O port they must use included in their address space (e.g., some RLL, ESDI, and SCSI disk controllers, and also some network cards) and shadowing is done using protected RAM. These ROMs will sometimes work when shadowed by this driver if they are within a protection block also containing UMBs. Try letting UMB_DRVR shadow the ROM and see if it works. If your system hangs up or you have problems with disk or network access with the ROM shadowed, you must use the /M= parameter to exclude it from UMB_DRVR.SYS control. Refer to CHIP-SET-SPECIFIC NOTES for any special considerations.

### DMA access upper memory

DMA (Direct Memory Access) is a method of data transfer between main memory (RAM) and I/O devices without requiring CPU intervention. Standard AT-compatible floppy disk controllers use it, as do some other devices, such as CD-ROMs and data acquisition hardware. It is supported through the Intel 8237A DMA controller chip, or by compatible integrated devices like the 82C206, or even by compatible components within the motherboard chip set. These devices control the data / address busses in the system to do the transfer while the CPU does other work. They do not have access to the internal 386+ CPU memory management facilities during their operation, and so are unaware of remapping of memory done there. Most of the time this is not a concern, as software that handles the remapping also handles DMA setup by intercepting accesses to the DMA controller registers and trying to keep things straight for DMA I/O. This can become a concern if the DMA transfer spans a page boundary that in virtual memory is to an adjacent page, but in real memory is not. Most of the 386-type mappers either automatically or through parameters try to avoid this situation by causing the area to be mapped (as much as possible) into contiguous memory.

UMB_DRVR, on the other hand, uses hardware external to the CPU that maps in the upper memory area contiguously such that DMA access to the upper memory area is no different than to the base 640K. The DMA mapping requirements that the device drivers are aware of for a standard AT system do not change when UMB_DRVR provides the upper memory area. Note that some drivers still cannot load high because they are "confused" by being at a higher address in memory than the program that is using them, but this is becoming much less common as drivers are rewritten to be able to take advantage of the DOS 5 high memory capabilities.

One consideration remains, however: *very* few chip sets that are supported by UMB_DRVR are designed such that the RAM that is mapped into the upper memory area can only be accessed by the CPU. UMB_DRVR performs a test for proper DMA function at initialization and issues a warning message if DMA is not possible to the upper memory area. If (and *only* if) this is the case, any DMA accesses attempted to an area above 640K and below 1M will not work, and the following restrictions will apply:

1. If you boot from a floppy disk or try to load any driver or TSR high reading it from a floppy, do not load UMB_DRVR.SYS — if you do, your system will probably hang up immediately upon trying to load anything into upper memory.
2. Almost no hard disk controllers use DMA, but if you have one that does, you will probably have problems with loading any driver or TSR into upper memory, and you may not be able to use UMB_DRVR at all.
3. Device drivers that use DMA for access to buffers within the driver itself, or allocated immediately after the driver when it initializes, cannot be loaded high on your system.
4. No DOS base memory expansion from unused video memory should be used (all DOS base RAM should be capable of DMA access).

***Note:*** These restrictions apply *only* if DMA access to upper memory is *not* available (i.e., if UMB_DRVR issues the warning message).

### Microsoft Windows

Windows 3.0 and 3.0A have been tested as follows:

386 Enhanced mode:

Windows will run in 386 enhanced mode with UMB_DRVR if the line

```
EMMExclude=A000-FFFF
```

is added to the SYSTEM.INI file, [386Enh] section. Note that a practical minimum of 4MB of RAM on your system is suggested to run in this mode. Also note that SETVER.EXE must be loaded LOW, and ANSI.SYS (if used) must be loaded LOW. There may be other drivers like this . . . experiment. Refer to UMB_EMS4.DOC for considerations regarding UMB_EMS4 and Windows.

**Standard mode:**

Windows in standard mode works with UMB_DRVR and HIMEM.SYS, with or without an EMS driver (hardware or software) loaded high or low. Note you must have a minimum of something like 512K or more extended memory to run in standard mode (i.e., do not have a software EMS driver remap ALL of your extended memory to expanded).

### Program access to upper memory with DOS 5 managing UMBs

1. Record current status of memory system so you can restore it.

   int 21H/ax=5800h - returns al=strategy (see below)

   int 21h/ax=5802h - returns al=UMB link state (see below)

2. Set up for memory allocation / deallocation.

   int 21h/ax=5801h/bh=0/bl=strategy

   int 21h/ax=5803h/bh=0/bl=UMB status

| | | |
|---|---|---|
| 00h | first fit, low memory | 00h = remove UMBs from mem chain |
| 01h | best fit, low memory | 01h = add UMBs UMBs to mem chain |
| 02h | last fit, low memory | (UMBs must be chained for access) |
| 40h | first fit, high memory | |
| 41h | best fit, high memory | |
| 42h | last fit, high memory | |
| 80h | first fit, try high and then low memory | |
| 81h | best fit, try high and then low memory | |
| 82h | last fit, try high and then low memory | |

3. Do normal int 21h/ah=48h, int 21h/ah=49h, int 21h/ah=4Ah as desired.
4. Restore values saved in step 1 above.

The system defaults to first-fit-low with UMBs not chained. If you chain the UMBs, strategies 00/01/02 affect the entire chain. For example, with the UMBs chained and strategy 00, you will get memory from the UMB area if the request cannot be satisfied from low memory.

### Determining your chip set

If your system documentation or CMOS setup screen does not tell you what chip set you have, the best way to find out is to open the cover on your system and look at the motherboard.

***Warning:*** MAKE SURE THE SYSTEM IS OFF AND UNPLUGGED WHEN YOU OPEN IT TO LOOK AT THE MOTHERBOARD!

The chip you will be looking for may not be one of the larger in size, but it will have many (usually 80+, sometimes up to 200 or more) pins. The number that identifies the key chip in the set is listed in CHIP-SET-SPECIFIC NOTES for each chip set supported. If you find a matching number on one of the chips on your motherboard, use the /C= value shown for that set. If you don't see a match, refer to the sections BAD NEWS, MAYBE?, and COMING ATTRACTIONS. Note that some chips only contain peripheral support and *do not* indicate what chip set you have. These include, but are not limited to

82C206 (many brands)
82C601, 82C710, 82C711 (Chips and Technologies)
VL82C100, VL82C106, VL82C107, VL86C050, VL16C45x, VL16C55x
(VLSI Technology)
82C452, 85C206 (Silicon Integrated Systems)
TACT82206 (Texas Instruments)

Again, these chip numbers *do not* indicate what chip set you have. If you find one of these on your motherboard, you should keep looking!

## Chip note specific notes

### User-specified available memory mode

/C=00 - Chip ID(s): None (286, 386SX, 386DX, 486)

This selection causes UMB_DRVR to map the areas specified in the /M= parameter, using the + (plus) character, as upper memory. Please note the following points when using this mode:

1. This mode *cannot be used* unless you have a way to actually map read-write RAM into the area between 640K (A0000) and 1M (100000) through your CMOS setup or some other program provided by your system or chip-set manufacturer, or a memory expansion card that maps RAM into that area on the AT bus. ***Note:*** The system BIOS brand MRBIOS from Microid Research will include the capability to set up your memory this way if your chip set supports it.
2. No verification or manipulation of any chip set is done.
3. ROM shadowing is totally controlled by your system BIOS and CMOS setup parameters.
4. No checking is performed other than the memory read-write and DMA tests.
5. Warm boots are *not* forced to be cold boots.
6. It is *your responsibility* to properly determine and specify which areas can be used.

### Chips & Technologies: CS8221 NEAT, CS8281 NEATsx, CS8223 LeAPset, CS8283 LeAPset-sx, Texas Instruments TACT82S411 Single Chip AT, United Microelectronics (UMC): UM82C210 286/386SX AT

/C=01 - Chip ID(s): (C & T) 82C212, 82C241 (286) 82C812, 82C841 (386SX); (TI) TACT82S411 (286, 386SX); (UMC) UM82C212 (286, 386SX)

These chip sets allow the 384K of motherboard RAM at A0000-FFFFF to relocate to extended memory at 100000-15FFFF on systems with only 1MB of RAM. If this remapping is enabled when UMB_DRVR.SYS enables this area, the remapping is removed and the size of your extended memory is reduced by 384K (i.e., it disappears). Note that this applies only to systems with *exactly* 1MB of memory. These chip sets map in 16K segments, but write protection for the area C0000-FFFFF is in 64K segments. To allow maximum memory utilization the driver marks any segment containing UMBs as read/write. If the 64K segment also contains a ROM shadow, it is not protected. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude any non-ROM areas within the 64K segment (e.g., for a VGA ROM at C0000-C7FFF, exclude C8000-CFFFF). If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF). UMB_DRVR.SYS will recognize the EMS setup for these chip sets and will exclude the EMS base segment if the EMS hardware is enabled when UMB_DRVR initializes. Use of the /M= parm is not required in this case.

### VLSI Technology VL82C200

/C=02 - Chip ID(s): VL82C201,VL82C202,VL82C203,VL82C204 (286, 386SX) (all 4 chips required)

This chip set uses a jumper or switch to enable shadow RAM ability. This does not actually cause shadowing, but must be in proper position for UMB_DRVR.SYS to work. Check your system documentation. This chip set maps and protects in 64K segments. To allow maximum memory utilization, 64K segments containing UMBs are set to read/write. If the 64K segment also contains a ROM shadow, it is not protected. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude any non-ROM areas within the 64K segment (e.g., for a VGA ROM at C0000-C7FFF, exclude C8000-CFFFF). If you must force a ROM to be unshadowed, exclude the entire 64K segment on a 64K boundary (e.g., for a disk ROM at C8000-CBFFF, exclude C0000-CFFFF).

### FOREX FRX36C300

/C=03 - Chip ID(s): FRX36C300 (386DX)

This chip set maps in 32K segments from C0000 to EFFFF, and one 64K segment for the system BIOS (F0000-FFFFF). RAM at A0000-BFFFF is always remapped to the highest area and cannot be used by the driver. Protection is set globally, meaning that all used RAM (shadow or UMBs) is marked read/write. The chip set also remaps D0000-EFFFF to the highest area if there is nothing shadowed in that area. When UMB_DRVR.SYS enables this area, the remapping is removed and the size of your extended memory is reduced by 128K. If you must force a ROM to be unshadowed, exclude the entire 32K segment on a 32K boundary (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CFFFF).

**Chips & Technologies: CS8230 386/AT, CS8231 Turbo Cache 386/AT, CS8233 PEAKset/386, CS82310 PEAKset DM/386**

/C=04 - Chip ID(s): 82C302, 82C307, 82C311, 82C351 (386DX)

These chip sets maps in 16K segments, but write protection for the area C0000-FFFFF is in 64K segments. To allow maximum memory utilization, the driver marks any segment containing UMBs as read/write. If the 64K segment also contains a ROM shadow, it is not protected. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude any non-ROM areas within the 64K segment (e.g., for a VGA ROM at C0000-C7FFF, exclude C8000-CFFFF). If you must force a ROM to be unshadowed, you need only exclude the 16kK segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

**Chips & Technologies: 82C235 SCAT, 82C836 SCATsx, CB8291 ELEAT, CB8295 ELEATsx, CS8285 PEAKset-sx, CS8227 CHIPSlite, CS8288 CHIPSlite-sx**

/C=05 - Chip ID(s): 82C235 (286), 82C836 (386SX)

These chip sets allow the 384K of motherboard RAM at A0000-FFFFF to relocate to extended memory at 100000-15FFFF on systems with only 1MB of RAM. If this remapping is enabled when UMB_DRVR.SYS enables this area, the remapping is removed and the size of your extended memory is reduced by 384K (i.e., it disappears). Note that this applies only to systems with *exactly* 1MB of memory. These chip sets map in 16K segments, but write protection for the area C0000-FFFFF is in 32K segments. To allow maximum memory utilization, the driver marks any segment containing UMBs as read/write. If the 32K segment also contains a ROM shadow, it is not protected. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude any non-ROM areas within the 32K segment (e.g., for a ROM at C8000-CBFFF, exclude CC000-CFFFF). If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### ETEQ Micro COUGAR II

/C=06 - Chip ID(s): 82C491 (386DX, 486)

This chip set maps in 16K segments from C0000 to EFFFF, and one 64K segment for the system BIOS (F0000-FFFFF). Memory protection is done in 64K segments from C0000 to EFFFF. The hardware does not allow read/write access to the area F0000-FFFFF (i.e., the ROM can be shadowed and protected, but the driver cannot use the boot page). The driver also cannot use the RAM at A0000-BFFFF. The chip set can remap A0000-BFFFF and D0000-EFFFF to the highest area if no shadowing is done in that area. If this remapping is enabled and UMB_DRVR.SYS enables the area D0000-EFFFF, the remapping is removed and the size of your extended memory is reduced by 256K. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude non-ROM areas within the 64K segment: e.g., for a ROM at C8000-CBFFF, exclude C0000-C7FFF and

CC000-CFFFF. If you have a VGA ROM at C0000-C7FFF, you only need exclude CC000-CFFFF. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### OPTi Sx/AT, Sx/AT Cache

/C=07 - Chip ID(s): 82C281, 82C282, 82C283 (386SX)

These chip sets maps in 16K segments from C0000 to EFFFF, and one 64K segment for the system BIOS (F0000-FFFFF). Memory protection is in 64K segments from C0000 to EFFFF. The hardware does not allow read/write access to the area F0000-FFFFF (i.e., the ROM can be shadowed and protected, but the driver cannot use the boot page). The driver also cannot use the RAM at A0000-BFFFF. These chip sets can remap A0000-BFFFF and D0000-EFFFF to the high end of extended memory if no shadowing is done in either area. If this remapping is enabled and UMB_DRVR.SYS enables the area D0000-EFFFF, the remapping is removed and the size of your extended memory is reduced by 256K. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude non-ROM areas within the 64K segment: e.g., for a ROM at C8000-CBFFF, exclude C0000-C7FFF and CC000-CFFFF. If you have a VGA ROM at C0000-C7FFF, you only need exclude CC000-CFFFF. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### OPTi DX/BB PC/AT

/C=08 - Chip ID(s): 82C496 (386DX, 486)

This chip set maps in 16K segments from C0000 to EFFFF, and one 64K segment for the system BIOS (F0000-FFFFF). Memory protection is done in 64K segments from C0000 to EFFFF. The hardware does not allow read/write access to the area F0000-FFFFF (i.e., the ROM can be shadowed and protected, but the driver cannot use the boot page). The driver also cannot use the RAM at A0000-BFFFF. The chip set can remap A0000-BFFFF and D0000-EFFFF to the highest area if no shadowing is done in that area. If this remapping is enabled and UMB_DRVR.SYS enables the area D0000-EFFFF, the remapping is removed and the size of your extended memory is reduced by 256K. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude non-ROM areas within the 64K segment: e.g., for a ROM at C8000-CBFFF, exclude C0000-C7FFF and CC000-CFFFF. If you have a VGA ROM at C0000-C7FFF, you only need exclude CC000-CFFFF. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### OPTi 386WB PC/AT, 486SXWB PC/AT

/C=09 - Chip ID(s): 82C391 (386DX), 82C493 (486)

These chip sets maps in 16K segments from C0000 to EFFFF, and one 64K segment for the system BIOS (F0000-FFFFF). Memory protection is done in 64K segments from C0000 to EFFFF. The hardware does not allow read/write access to RAM at F0000-FFFFF (i.e., the ROM can be shadowed and protected, but the driver cannot use the boot page). The driver also cannot use the RAM at A0000-BFFFF. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude non-ROM areas within the 64K segment: (e.g., for a ROM at C8000-CBFFF, exclude C0000-C7FFF and CC000-CFFFF). If you have a VGA ROM at C0000-C7FFF, you only need exclude CC000-CFFFF. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### OPTi 386/486WB EISA

/C=10 - Chip ID(s): 82C682 (386DX, 486)

This chip set maps and protects in 16K segments at C0000-DFFFF, one 64K segment at E0000 and one 64K segment for the system BIOS at F0000. The driver cannot use the RAM at A0000-BFFFF. If one of the 64K segments contains both shadowed ROM and UMB area, it is marked read/write. All ROMs in the C0000-DFFFF area are protected. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF) unless it is in the E0000 block or you wish to force the BIOS (F0000-FFFFF) to be unshadowed. Then you must exclude the entire 64K block (E0000-EFFFF and/or F0000-FFFFF).

### Elite Microelectronics Eagle, Falcon

/C=11 - Chip ID(s): e88C311 (386DX), e88C411 (486)

These chip sets map and protect in 16K segments for the entire area C0000-FFFFF. All shadowed ROMs are write-protected. UMB_DRVR cannot use the RAM at A0000-BFFFF. These sets always remap A0000-BFFFF to the highest memory area, and can selectively remap C0000-FFFFF in 64K blocks if no shadowing is done within the 64K block. If this remapping is active and UMB_DRVR enables shadow memory within one of the remapped 64K blocks, the remapping is removed and the size of your extended memory is reduced. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### VLSI Technology SCAMP

/C=12 - Chip ID(s): VL82C310, VL82C311 (286, 386SX), VL82C311L (286)

These chip sets map and protect in 32K segments for A0000-BFFFF, 16K segments for C0000-DFFFF, and 32K segments for E0000-FFFFF. If a ROM shadow shares a 32K segment from E0000 to FFFFF with a UMB area, it is marked read/write. Any shadowed ROM from C0000-DFFFF is protected, as is any 32K segment from E0000 to FFFFF that is all shadowed ROM. These sets can remap A0000-FFFFF to the highest memory area if no shadowing is done and system memory is 1MB, 2MB, 3MB, or 4MB. If remap is active and UMB_DRVR enables any shadow memory, the remapping is removed and the size of your extended memory is reduced. Note that this applies only to systems with *exactly* 1MB, 2MB, 3MB or 4MB installed. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies if it is between C0000 and DFFFF (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF). If it is between E0000 and FFFFF, exclude all areas in the 32K segment (e.g., for a disk ROM at E0000-E3FFF, exclude E0000-E7FFF).

### VLSI Technology: VL82C286, VL82C386 TOPCAT, Intel 82340SX, 82340DX, (These sets are all made by VLSI Technology)

/C=13 - Chip ID(s): (VLSI) VL82C320 (286, 386SX), VL82C330 (386DX)

VL82C320A (286, 386SX, 486)

(Intel) 82343, 82346 (286, 386SX)

82343A (286, 386SX, 486)

These chip sets map in 16K segments for the entire area from A0000 to FFFFF and protect in 16K segments from C0000 to FFFFF. All shadowed ROM areas are protected. The video area (A0000-BFFFF) and the boot page (F0000-F7FFF) can only be utilized on the VL82C320 / 82343 "A" revisions (this implementation was chosen to avoid the DMA limitation; see MISCELLANEOUS NOTES — DMA ACCESS TO UPPER MEMORY). UMB_DRVR will recognize the various chips and enforce these restrictions accordingly. These sets can remap A0000-FFFFF to the high end of extended memory if no shadowing is done and system memory is exactly 1MB or 2MB. If this remapping is active and UMB_DRVR enables shadow memory, the remapping is removed and the size of your extended memory is reduced. Note that this applies only if the system memory size is exactly 1MB or 2MB. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### OPTi HiD/386 AT, HiB/486 AT

/C=14 - Chip ID(s): 82C382 (386DX), 82C482 (486)

These chip sets map in 16K segments from C0000 to EFFFF and one 64K segment for the system BIOS (F0000-FFFFF). Memory protection is done in 64K segments from C0000 to EFFFF. The hardware does not allow read/write access to the area F0000-FFFFF (i.e., the ROM can be shadowed and protected, but the driver cannot use the boot page). The driver also cannot use the RAM at A0000-BFFFF. The chip sets can remap A0000-BFFFF and D0000-EFFFF to the highest area if no shadowing is done in that area. If this remapping is enabled and UMB_DRVR.SYS enables the area D0000-EFFFF, the remapping is removed and the size of your extended memory is reduced by 256K. Although it should not be necessary, if you wish to have a ROM be shadowed and protected, exclude non-ROM areas within the 64K segment: e.g., for a ROM at C8000-CBFFF, exclude C0000-C7FFF and CC000-CFFFF. If you have a VGA ROM at C0000-C7FFF, you only need exclude CC000-CFFFF. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### Sun Electronics SUNTAC ST62CS24, ST62CS25

/C=15 - Chip ID(s): ST62C241 (286), ST62C251 (286, 386SX)

These chip sets have two memory-mapping modes: one for extended memory and one for expanded (EMS) memory. You must have your system configured for extended memory only for UMB_DRVR to recognize the chip set. Also, some BIOSs (e.g., newer AMI) relocate the video ROM before shadowing it. This can cause fragmentation of your upper memory. If this is the case, turn off video shadowing in your CMOS / extended setup. The driver will shadow your video ROM when it sets up the upper memory area. See your CMOS setup or system documentation. These chip sets map and protect in 16K segments from C0000 to DFFFF and in 32K segments from E0000 to FFFFF. The driver cannot use the RAM at A0000-BFFFF. All shadowed ROMs from C0000 to DFFFF and the system BIOS are protected. The chip sets always remap A0000-DFFFF to the highest area, and remap E0000-FFFFF there if no shadowing is done. UMB_DRVR must remove the remapping for E0000-FFFFF if active, and must use some extended memory to supply the upper memory area. Your extended memory size will be adjusted accordingly. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### Texas Instruments TACT83000

/C=16 - Chip ID(s): TACT83442 (386SX, 386DX, 486)

This chip set maps in 16K segments from A0000 to FFFFF, and memory protection is done in 16K segments from C0000 to FFFFF. All shadowed ROMs are protected. The chip set can remap any 64K segment from A0000 to FFFFF not used for shadowing to the upper end of extended memory. If this remapping is active and UMB_DRVR.SYS uses any of the remapped area, the remapping is removed and the size of your extended memory is reduced. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### Silicon Integrated Systems High Performance 80386

/C=17 - Chip ID(s): 85C310 (386DX)

This chip set maps in 16K segments from C0000 to DFFFF, and two 64K segments for E0000 and F0000. Memory protection is global, so no shadow areas are protected. The driver cannot use the area from A0000 to BFFFF. The chip set, depending on memory configuration, can remap either 256K or 384K of unused shadow area to the upper end of extended memory. If this remapping is active and UMB_DRVR uses any of the remapped area, the remapping is removed and your extended memory size is reduced. If you must force a ROM to be unshadowed, you need only exclude the 16K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CBFFF).

### Silicon Integrated Systems High Performance ISA486

/C=18 - Chip ID(s): 85C401 (486)

This chip set maps in 32K segments from C0000 to EFFFF, and one 64K segment for F0000. Memory protection is global, so no shadowed areas are protected. The driver cannot use the area from A0000 to BFFFF. If you must force a ROM to be unshadowed, you must exclude the 32K segment(s) it occupies (e.g., for a disk ROM at C8000-CBFFF, exclude C8000-CFFFF).

## Installation

***Warning:*** Please be sure you have reviewed the miscellaneous and chip-set-specific notes prior to installing. Also, make sure you have backed up your system and that you have a diskette you can boot from in case you have problems with your CONFIG.SYS settings.

Installation (preferably as the first driver) is via the lines:

```
DEVICE=UMB_DRVR.SYS /C=nn
   [/M=ssssssssssssssssssssssss]

DOS=[HIGH|LOW],UMB (Required -
   turn on DOS 5 UMB support)
```

in your CONFIG.SYS file. The /C= parameter is required — nn is the chip set from "CHIP-SET-SPECIFIC NOTES." The /M= parameter is optional and is used to override defaults.

## Error messages

```
DMA NOT SUPPORTED (WARNING)
```

See MISCELLANEOUS NOTES — DMA ACCESS TO UPPER MEMORY.

```
SHADOW RAM TEST FAILURE
```

This message is issued when the shadow RAM read/write test fails. It is usually an indication that you are trying to use memory that is not available, or (possibly) you do not have the chip set you have specified. Check your CMOS setup and any jumpers or switches per your hardware documentation. Also review CHIP-SET-SPECIFIC NOTES earlier in this document for any requirements. This can also occur if you are trying the example listed in the INSTALLATION section using a portion of the video RAM area on a VGA system in CGA mode and your VGA hardware will not allow it.

```
CHIP SET NOT RECOGNIZED
```

As much as possible, UMB_DRVR.SYS tries to verify that you have the chip set you indicated in the /C= parameter. If you are sure you have the chip set and have correctly specified it, contact me (see TECHNICAL SUPPORT) and I will try to straighten things out. This message can also occur if a program (e.g., CMOS setup from DOS, disk defragger) reboots the system in a certain way. In this case, it can be cleared with the reset button or by power down / power up.

```
XMS ALREADY INSTALLED
```

You have not installed UMB_DRVR.SYS before HIMEM.SYS (UMB_DRVR.SYS issues message), or you have omitted or incorrectly specified the line **DOS=xxxx,UMB** in your CONFIG.SYS (HIMEM.SYS issues message). Correct your CONFIG.SYS file and reboot.

```
INCORRECT DOS VERSION
```

UMB_DRVR.SYS requires MS/DOS 5.0 (or higher) for proper operation.

```
INVALID PARAMETER(S)
```

On the DEVICE= statement for UMB_DRVR.SYS you have: 1) omitted or incorrectly specified the /C= parameter, 2) incorrectly specified the /M= parameter, or 3) included extra parameter(s). Check that you have entered the proper 2-digit number for your chip set, that (if specified) the /M= parameter contains 24 characters from the set "." (period), '–' (minus), '+' (plus), and '#' (pound sign), and that nothing else is specified. Correct your CONFIG.SYS file and reboot.

Please see the file UMB_DRVR.DOC for details on the following topics which have been omitted here:

1. BAD NEWS (CHIPS THAT WILL NOT BE SUPPORTED)
2. MAYBE? (NEED TECHNICAL DATA TO SUPPORT)
3. COMING ATTRACTIONS (SUPPORT PLANNED OR UNDER DEVELOPMENT)

If your chip set is not listed, have your system vendor or the chip set manufacturer contact me and I will try to support it.

A newer, more flexible (read as less Neanderthal, approaching the Bronze Age) user interface is coming as soon as I get the time.

Also in the works are features to save even more precious memory below 640K by:

- — loading the primary shell (COMMAND.COM) into upper memory
- — relocating all DOS areas possible to upper memory, including FCBS=, FILES=, BUFFERS=, STACKS=, and LASTDRIVE=
- — allowing the lower portion of the video RAM area to be switched in and out to expand DOS base RAM beyond 640K but not inhibit graphics modes (only for chip sets with video area support)

I have (I think) found a way to *reliably* test for and list the chip set in a machine. I will be including a separate program to do this in the package soon. Of course, it will only recognize the chip sets it supports, so a negative result will not necessarily mean you have a chip set that will not be supported later.

If there is enough interest, I will also write device-specific EMS drivers for the hardware facilities in the EMS-capable chip sets. Please Emily or surface mail your thoughts (no phone calls on this, please — save those for support questions).

## Technical Support

Technical support, including preregistration questions or install assistance, is available by phone at your expense. Please be aware that I am in the Eastern US time zone (GMT-4 or GMT-5 depending on season) and try to call at a reasonable hour: that is, 9 a.m. to Noon, 1 p.m. to 5 p.m., or 7 p.m. to 10 p.m. Saturday is OK, but *please* avoid Sundays. I can also be contacted via Emily on CompuServe, BIX, and INTERNET — I monitor my mail almost every day. It is not necessarily a good idea to leave me messages on CompuServe in the forum sections unless your question or discussion is of general interest. The Postal Service may also be used.

# UMB_EMS4.SYS Device Driver

**Version 5.22**
**EMS simulator for 286 / 386 / 486 systems**
**Copyright © 1991, 1992 by Christopher Blum**

## Introduction

UMB_EMS4.SYS is a device driver providing EMS simulation from XMS (HIMEM.SYS) extended memory. It is a software implementation of the LIM 4.0 standard, implementing all features of the specification except page aliasing, which requires actual hardware support.

All XMS extended memory is considered to be available also as EMS. The XMS memory is dynamically configured to EMS and back as programs open and close EMS handles. You should see, when you do a MEM display, that the EMS available and XMS available track with each other.

***Important:*** The driver must be installed *after* UMB_DRVR.SYS and HIMEM.SYS have been installed, preferably using the DEVICEHIGH command to load into upper memory. It requires about 72K of memory to initialize, and takes no parameters.

## Requirements and Restrictions

UMB_EMS4 has been run on 286, 386, and 486 systems with excellent results. It will run on any system supported by UMB_DRVR.SYS, DOS 5.0, and the HIMEM.SYS XMS server.

Do *not* use simulated EMS provided by UMB_EMS4 for disk cache programs such as PC-Quick, HyperDisk, SMARTDRV.SYS, etc. They will work better with extended memory than simulated EMS (see also following).

The driver is Windows-386-enhanced-mode-aware, but you need to start Windows in enhanced mode using **WIN /3** (UMB_EMS4 will back itself out at Win init and reinstall at Win exit, but you must force enhanced mode) if you want it. The driver will not allow Windows to start in enhanced mode if any simulated EMS users are active. This is to allow Windows full use of extended memory, and further explains why you can't have any cache programs use EMS if they need to run while Windows is active. This can also affect shell programs or COMMAND.COM replacement programs such as 4DOS. If you have a choice, select to use extended memory for any such applications.

I *strongly* suggest that you set your XMS handle count to 64 by using /NUMHANDLES=64 for HIMEM.SYS, because the driver uses XMS handles and the EMS 4.0 specification says you should have a minimum of 64. It will work with the HIMEM.SYS default of 32, but there is some chance that programs which normally would run might not in that environment.

## Miscellaneous Notes

If you're interested, here are some benchmark figures, run on a 16 MHz 386sx with Chips & Technologies NEATsx chip set with 4MB memory. The hardware figures below reflect the NEATsx EMS facilities using Chips and Technologies NEAT EMS driver version 2.0 and 2MB EMS.

**PC Magazine Labs Benchmark 6.0**

| | *Hardware* | *UMB_EMS4* |
|---|---|---|
| EMS read | 527.83 | 12.80 (K/sec) |
| EMS write | 527.83 | 12.80 |

**MOVETEST (PC Labs EMS Move test, 256KB move timing in seconds)**

— 64 byte records — 128K records —

In the following table, **Conv** means **Conventional**.

| | *Hardware* | *UMB_EMS4* |
|---|---|---|
| EMS => Conv | 1.81 | 1.42 |
| Conv => Conv | 0.71 | 0.65 |
| Conv => EMS | 1.86 | 1.42 |
| EMS => EMS | 1.86 | 1.64 |

| | *Hardware* | *UMB_EMS4* |
|---|---|---|
| EMS => Conv | .05 | .05 |
| Conv => Conv | .05 | .05 |
| Conv => EMS | .05 | .05 |
| EMS => EMS | .05 | .05 |

**Disk defragmentation program**

(100MB drive, 80MB data. Program run two times, organizing data in two different orders. Identical initial data for all runs. Program used 2MB of EMS or Extended memory for buffers)

| | *Run #1* | *Run #2* | *Avg* |
|---|---|---|---|
| | *(times in min:sec)* | | |
| EMS (Hardware) | 7:03 | 7:15 | 7:09 |
| EMS (EMM386) | 7:15 | 7:32 | 7:24 |
| Extended (XMS) | 8:53 | 9:14 | 9:04 |
| EMS (UMB_EMS4) | 11:50 | 11:05 | 11:28 |

Don't let these figures scare you — most programs tested run very well with UMB_EMS4 and show little performance difference. The above figures are *worst case* and do not reflect the average program. Many programs such as Quattro Pro 3.0, Lotus, dBASE, Paradox, a selection of games, several TSRs including a print spooler, and various others have been tested with no problems.

## Installation

***Important:*** Please be sure you have reviewed this documentation prior to installing. Also, make sure you have backed up your system and that you have a diskette you can boot from in case you have problems with your CONFIG.SYS settings.

Installation is via the lines

```
DEVICE=UMB_DRVR.SYS ...
DOS=HIGH,UMB
DEVICEHIGH=HIMEM.SYS /NUMHANDLES=64
DEVICEHIGH=UMB_EMS4.SYS
```

The driver will print a status report at initialization, and self-explanatory error messages if installation fails.

If you do not have enough upper memory to load UMB_EMS4 high, you may use DEVICE= to load into conventional memory, but it is much better to load it high if possible. Rearranging your CONFIG.SYS to do so is highly suggested.

Again, as with any new software, *make sure that your system is backed up,* and verify the operation of your programs. It is impossible for me to test all configurations and programs. If you have problems, let me know and I will try to straighten them out.

## Known Incompatibilities

WordPerfect version 5.1 crashes if you use its default EMS 4.0 support. This can be avoided by either using the execute command

```
WP /32
```

when running it, or the command

```
SET WPC=/32
```

in the AUTOEXEC.BAT file at bootup. Either of these actions tells WordPerfect to use LIM 3.2 calls only. I am in contact with WordPerfect Corporation and am working on the problem.

## UMB_DRVR.SYS, Version 5.22 Registration Form

Name: ____________________
Company: ____________________
Address: ____________________
City: ____________________
State: ________ Country: ____________
Zip: ________ Phone: ____________

**Remit to:**
Christopher Blum
1022 East Wayne Avenue
Wooster, Ohio 44691

(216) 262-3786
CompuServe: 76625,1041
INTERNET: 76625.1041@compuserve.com
BIX: cblum

* Email system / routing / ID ____________
Computer description, configuration, chip set, etc.
____________________
____________________
____________________
____________________
____________________

Comments ____________________
____________________
____________________
____________________
____________________

Cost (U.S. Funds) $25 (no credit cards).

# VIRUSCAN

**Version 9.13V101**
**Copyright © 1989–1993 by McAfee Associates**

## System Requirements

SCAN requires 320K of RAM and DOS 2.0 or above (some features require DOS 3.1 or above).

SCAN is designed to check stand-alone and networked PCs for viruses. For network file servers please use NETSCAN instead.

SCAN displays messages in English (default); foreign language support is available as well.

## Program and Documentation Files

By default, the install program for the *DOS 6 SECRETS* diskettes will install all of the VIRUSCAN files in the directory \SECRETS\SCAN. This directory has not been added to the search path. You can execute the program from its home directory, add the directory to the search path, or copy the VIRUSCAN files to a directory that is included in the search path.

***Important:*** Write-protect the floppy disk containing the VIRUSCAN (SCAN.EXE) program before scanning to prevent VIRUSCAN from becoming infected by a computer virus.

## Overview: (Known Virus Detection)

VIRUSCAN Version 9.13V101 (filename SCAN.EXE) identifies all 1,134 known computer viruses and their variants. Some viruses have been modified so that more than one "strain" exists. Counting such modifications, 1,830 viruses exist.

All known viruses infect one or more of the following areas: the hard disk partition table (alias Master Boot Record); the DOS Boot Sector of disks; or one or more executable files on the system. Executable files include operating system files, .COM files, .EXE files, overlay files, or any other files containing program code. A virus that infects more than one area, such as a boot sector and an executable file is called a multipartite virus.

SCAN checks files, subdirectories, diskettes, or entire systems for pre-existing computer virus infections. It will identify the virus infecting the system and the area where it was found, giving the name of the virus as well as the I.D. code used with CLEAN-UP to remove it.

Infected files can be removed using the /D switch in SCAN to erase the file, or with the CLEAN-UP universal virus removal (disinfection) program. CLEAN-UP is recommended because in most cases it will eliminate the virus and fully restore infected programs or system areas to normal operation.

The VIRLIST.TXT file describes all viruses identified by SCAN and their associated I.D. codes for removal by CLEAN-UP.

## Overview: (Unknown and New Virus Detection)

SCAN has three separate methods of detecting unknown and new viruses:

- Validation codes which can be periodically checked against to look for the changes made by a virus to files or system areas.
- Generic and Family virus detectors to look for new viruses which are derivatives of older viruses.
- External virus signatures to insert new virus signature strings on a temporary basis to SCAN.

## Synopsis: (Technical Description of Known Virus Detection)

SCAN detects known viruses by searching the system for strings (sequences of bytes) unique to each computer virus and reporting their presence if found. For viruses which encrypt or cipher their code so that every infection of the virus is different, SCAN uses detection algorithms (programs) that work by statistical analysis, heuristics, or code disassembly.

## Synopsis (Technical Description of New/ Unknown Virus Detection)

SCAN checks for new or unknown viruses by comparing files against previously recorded validation (checksum) data. SCAN has two levels of validation which are stored in three separate ways:

- A simple 10-byte long validation checksum may be appended to .COM and .EXE files. If a file has been modified, it no longer matches the checksum and SCAN will report the file may have become infected. (/AV, /CV, /RV switches)
- An enhanced 52-byte validation and recovery data checksum can also be created. This can be appended to the end of files like the 10-byte checksum, or stored in a separate log file which can be offline (e.g., on floppies) for recovery purposes. CLEAN-UP can restore infected files, partition tables, or boot sectors using this information. (/AG, /CG, /RG switches and /AF, /CF, /RF switches)

*Note:* If validation codes are attached to files, SCAN will NOT add codes to the partition table, boot sector, or system files. Instead, a separate hidden file will be created in the root directory named SCANVAL.VAL containing data for these areas.

*Note:* Files which are self-checking (e.g., Lotus 1-2-3) should not be validated with the /AV (Add Validation) or /AG (Add Generic) switches which modify files. Instead, use the /AF (Add File) switch.

- SCAN also checks for new or unknown viruses by searching for Generic or Family virus strings. These are strings that have been found repeatedly in different viruses. Since virus writers may use the older pieces of code for new viruses, this allows SCAN to detect viruses which have not been written yet.

VIRUSCAN can be updated to search for new viruses by an External Virus Data File, which allows the user to input new search strings for viruses. (/EXT switch)

## Authenticity

Before using SCAN for the first time, verify that it has not been tampered with or infected by a virus by using the enclosed VALIDATE program. For instructions on using VALIDATE, please read the VALIDATE.DOC file.

The validation results for Version 9.13V101 should be

```
FILE NAME: SCAN.EXE
SIZE: 111,681
DATE: 02-11-1993
File Authentication
Check Method 1: B9BD
Check Method 2: 19EE
```

If your copy of SCAN differs, it may have been damaged or have options stored in it with the /SAVE switch. Run SCAN with only the /SAVE option to remove any stored options and then re-run VALIDATE. Always obtain your copy of VIRUSCAN from a known source. The latest version of VIRUSCAN and validation data for SCAN.EXE can be obtained from McAfee Associates' bulletin board system at (408) 988-4004 or from the McAfee Virus Help Forum on CompuServe (GO MCAFEE), or the mcafee.COM anonymous ftp site on the Internet.

VIRUSCAN performs a self-check when run. If SCAN has been modified in any way, a warning will be displayed and the user will be prompted to either continue or quit. SCAN can still check for viruses; however, if SCAN reports that it has been damaged, it is recommended that a clean copy be obtained.

Beginning with Version 72, all of McAfee Associates' VIRUSCAN series are archived with PKWare's PKZIP Authentic File Verification. If you do not see an -AV after every file is unzipped and receive the `Authentic Files Verified! # NWN405 Zip Source: McAFEE ASSOCIATES` message when you unzip the files, then do not use them. If your version of PKUNZIP does not have verification ability, then this message may not be displayed. Please contact us if you believe tampering has occurred to the .ZIP file.

## Command Summary

VIRUSCAN checks files and other areas of the system that can contain a computer virus. When a virus is found, SCAN identifies the virus and the file or system area where it was found.

SCAN examines files based on their extension. The default extensions supported by SCAN are .APP, .BIN, .COM, .EXE, .OV?, .PGM, .PIF, .PRG, .SWP, .SYS, and .XTP. Additional extensions can be added with the /E option, or use the /A to check all files.

**Command Format:**

```
SCAN {drive(s)} {options}
```

*{drive(s)}* — Indicates a drive or drives to be scanned

*Options* are

| | |
|---|---|
| \ | Scan root directory and boot area only |
| /? /H or /HELP | Displays help screen |
| /A | Scan all files, including data, for viruses |
| /AD | Scan all local drives for viruses |
| /AF {filename} | Store recovery & validation data to {filename} |
| /AG {filename} | Add recovery & validation data to files EXCEPT for those listed in {filename} |
| /AV {filename} | Add validation codes to files EXCEPT for those listed in {filename} |
| /BELL | Beep whenever a virus is found |
| /CERTIFY | List files that do not have a validation code |
| /CF {filename} | Check for viruses using recovery & validation data stored in {filename} |
| /CHKHI | Check memory from 0K to 1,088K |
| /CG | Check recovery & validation data on files |
| /CV | Check validation codes on files |
| /D | Overwrite and delete infected files |
| /DATE | Save the date and time SCAN was last run (use /SHOWDATE to display) |
| /E .xxx .yyy | Scan overlay extensions .XXX and .YYY |
| /EXT {filename} | Scan using external virus data from {filename} |

| | |
|---|---|
| /FAST | Speed up VIRUSCAN's output (see below for specifics) |
| /HISTORY {fname} | Create infection log {fname} appending to old log |
| /M | Scan memory for all viruses (see below for specifics) |
| /MAINT | Scan "invalid media" error (damaged) disk |
| /MANY | Scan multiple disks |
| /NLZ | Skip internal scan of LZEXE-compressed files |
| /NOBREAK | Disable Ctrl-C and Ctrl-Brk during scanning |
| /NOEXPIRE | Do not display expiration notice |
| /NOMEM | Disable memory check |
| /NOPAUSE | Disable screen pause when scanning |
| /NPKL | Skip internal scan of PKLITE-compressed files |
| /REPORT {fname}t | Create infection log {fname} deleting the old log |
| /RF filename | Remove recovery & validation data stored |
| /RG | Remove recovery & validation data from files |
| /RV | Remove validation codes from specified files |
| /SAVE | Save specified options as new default options |
| /SHOWDATE | Display the date and time SCAN was last run (use /DATE to save date and time) |
| /SUB | Scan all subdirectories inside a subdirectory |
| /UNATTEND | Scan using DOS critical error handler (required if used within Windows or OS/2) |
| @{filename} | Scan using options from {filename} |

Please see the file SCAN101.DOC for details on Options.

## Examples

The following examples show different option settings:

```
SCAN C:
```

To scan drive C:

```
SCAN A:R-HOOPER.EXE
```

Scans file "R-HOOPER.EXE" on drive A:

```
SCAN A: /A /CV
```

Scans all files and checks validation codes for unknown viruses on drive A:.

```
SCAN B: /D /A
```

Scans all files on drive B: and prompts for erasure of any infected files, if found.

```
SCAN C: D: E: /AV /NOMEM
```

Scan for viruses, add validation codes to files on drives C:, D:, and E:, and skip memory checking.

```
SCAN C: D: /M /A
```

Scan memory for all viruses, all files on drives C: and D: for viruses.

```
SCAN C: D: /E .WPM .COD
```

Scans drives C: and D:, including .WPM and .COD files

```
SCAN C: /EXT A:SAMPLE.ASC /BELL
```

To scan drive C: for known computer viruses and also for viruses added by the user via the external virus data file option, and beep whenever a virus is found.

```
SCAN C: /M /NOPAUSE /REPORT
A:INFECTN.RPT
```

To scan for all viruses in memory and drive C: without stopping, and create a log file INFECTN.RPT on drive A:

```
SCAN E:\DOWNLOADS /SUB
```

To scan all subdirectories under the directory DOWNLOADS on drive E:

```
SCAN C: D: E: /FAST /CERTIFY
```

To perform a fast scan of drives C:, D:, and E: and check for any files that do not have validation codes.

```
SCAN @C:\SCANOPTN.LST
```

To run SCAN using configuration file SCANOPTN.LST located in the root directory of drive C:.

```
SCAN /AD /M
```

Scan all hard drive partitions and memory for viruses.

## Exit Codes

After VIRUSCAN has finished running, it will set the DOS ERRORLEVEL. ERRORLEVELs are used in batch files to pass the results of a program's actions. The ERRORLEVELs returned by SCAN are:

| *Errorlevel* | *Description* |
|---|---|
| 0 | No viruses found |
| 1 | One or more viruses found |
| 2 | Abnormal termination (program error) |
| 3 | One or more uncertified files found |
| 4 | Ctrl-C or Ctrl-Break aborted scan |

If a user stops the scanning process, SCAN will set the ERRORLEVEL to 4. If you wish to prevent users from stopping the scanning process, then run SCAN with the /NOBREAK option.

## Virus Removal

What do you do if a virus is found? You can contact McAfee Associates for help, their authorized agents, or use the CLEAN-UP program.

McAfee Associates can be reached by BBS, CompuServe, FAX, Internet, or Telephone and there is no charge for support calls to McAfee Associates (Authorized agents may charge normal McAfee Associates consulting rates).

The CLEAN-UP universal virus disinfection program can disinfect virtually all reported computer viruses. It is updated with each release of the SCAN program to remove new viruses. CLEAN-UP can be downloaded from McAfee Associates' BBS, the McAfee Virus Help Forum on CompuServe, and the mcafee.COM and WSMR-SIMTEL20.Army.Mil sites on the Internet, or from any of the agents' BBSes listed in the enclosed AGENTS.TXT text file.

It is strongly recommended that you get experienced help in dealing with viruses if you are unfamiliar with anti-virus software and methods. This is especially true for *critical* viruses and partition table/boot sector-infecting viruses as improper removal of these viruses can result in the loss of all data and the use of the infected disk(s). (For a listing of critical viruses, see the /M switch listed under OPTIONS above).

Before removing a boot sector or partition table-infecting virus, it is recommended that you cold boot the infected PC from a clean DOS disk and back up any critical data.

For qualified assistance in removing a virus, contact McAfee Associates directly or any of the Authorized Agents in your area. Agents may charge McAfee Associates' normal consult rates for their services.

If you wish to remove a file-infecting virus manually, cold boot the PC from a clean (virus-free) DOS system disk and run SCAN with the /A and /D switches to erase all infected files. Any files removed in this manner cannot be recovered.

## Tech Support

McAfee Associates can be contacted by BBS, CompuServe, FAX, or InterNet 24 hours a day, or by telephone at (408) 988-3832, Monday through Friday, 7:00 a.m. to 5:30 p.m. Pacific Time.

## BBS Access

Our 25-line BBS is accessible 24 hours a day, 365 days a year, except for scheduled downtime and maintenance. All lines run US Robotics Courier HST Dual Standard ASL modems operating from 1,200 bps to 16,800 bps with line settings of 8 data bits, no parity, and one stop bit.

## The McAfee Virus Help Forum On CompuServe

We are now sponsoring the McAfee Virus Help Forum on CompuServe. To reach the McAfee

Virus Help Forum, type GO MCAFEE at any CompuServe prompt. A free introductory membership is available. For more information, please read the enclosed COMPUSER.NOT file.

## Internet Access to McAfee Asociates Software

The latest versions of McAfee Associates' anti-viral software is now available by anonymous ftp (file transfer protocol) over the Internet from the site mcafee.COM. If your domain resolver does not support names, use the IP# 192.187.128.1. Enter "anonymous" for your user I.D. and your own email address for the password. Programs are located in the pub/antivirus directory. If you have any questions, please send email to support@mcafee.COM

McAfee Associates' anti-viral software may also be found at the Simtel20 archive site WSMR-SIMTEL20.Army.MIL in the PD1:<MSDOS.TROJAN-PRO> directory and its associated mirror sites WUARCHIVE.WUSTL.EDU (US), NIC.SWITCH.CH (Swiss), NIC.FUNET.FI (Finland), SRC.DOC.IC.AC (UK), and RANA.CC.DEAK.OZ.AU (Australia).

Please see the file SCAN101.DOC for Appendixes A, B, and C.

## Important Notice — Please Read!

Due to the nature of anti-virus software, the slight chance exists that a virus may be reported in a file that is not infected by that virus.

If you receive a report of a virus infection which you believe may be in error, please contact McAfee Associates by telephone at (408) 988-3832, by fax at (408) 970-9727, or upload the file to our BBS at (408) 988-4004 along with your name, address, daytime telephone number, and electronic mail address, if any.

### VirusScan Registration Form

Name: ______________________________

Company: ___________________________

Address: ____________________________

City: _______________________________

State: __________ Country: ______________

Zip: __________ Phone: ________________

**Remit to:**
McAfee Associates, Inc.
3350 Scott Blvd. Bldg. 14
Santa Clara, CA 95054-3107

(408) 988-3832 office
(408) 970-9727 fax
(408) 988-4004 BBS (25 lines)
U.S.A. USR HST/v.32/v.42bis/MNP1-5
CompuServe GO MCAFEE
InterNet support@mcafee.COM

A registration fee of US $25 is required for the use of VIRUSCAN by individual home users. Registration entitles the holder to unlimited free upgrades from McAfee Associates' BBS, the Internet, and the McAfee Virus Help Forum on CompuServe as well as technical support for one year. When registering, a diskette containing the latest version may be requested for an additional US $9. Only one diskette mailing will be made.

Registration is for home users only and does not apply to businesses, corporations, organizations, government agencies, or schools, which must obtain a license for use. Contact McAfee Associates directly or an Authorized Agent for more information.

# Part VI: Appendixes

Page 809 ◊ **Appendix A**
Installing DOS 6

Page 815 ◊ **Appendix B**
DOS Command Quick Reference

Page 825 ◊ **Appendix C**
ANSI.SYS Quick Reference

Page 831 ◊ **Appendix D**
Interacting with Workgroups and Networks

Page 879 ◊ **Appendix E**
Additional DOS Information Sources

# Appendix A

# Installing DOS 6

## Upgrading to DOS 6

Upgrading to DOS 6 couldn't be easier, but before you execute the SETUP program from disk 1 of DOS 6, you need to make the following preparations:

- During the installation procedure, you will be prompted to insert an Uninstall disk. (SETUP will transfer old DOS files to this disk to enable a subsequent UNINSTAL, if it becomes necessary.) Have ready a disk that is the same media type as the DOS 6 disks and label the disk *Uninstall*. The disk does not need to be formatted, but if the disk is formatted and contains files, the files will be overwritten. If you are installing DOS from 360K disks, you will need two Uninstall disks.
- DOS 6 uses considerable more disk space than its predecessors, especially if you install the optional Windows utilities. Make sure that you have enough free space — between 5 and 10 megabytes on your boot drive. SETUP will check your available space and report an error if there is insufficient room on the boot drive.
- If you are running on a network, suppress mail and print notifications or log off the network. SETUP sometimes aborts when a memory-resident utility pops up a message during installation.
- Disable any third-party disk cache software.
- Disable any delete-protection software such as MIRROR.
- Disable any anti-virus TSRs that monitor disk writes.

Having made the necessary preparations, just insert disk 1 of the DOS 6 set into a disk drive (assumed to be drive A in this appendix's examples) and enter the following command:

```
a:setup
```

If you find that the display is difficult to read or even illegible, press F5. If this action doesn't help, press F3 twice to exit SETUP and then reenter the command using the /I switch to instruct SETUP not to perform any hardware checks:

```
a:setup /i
```

SETUP will then guide you through the DOS 6 installation process.

If you experience any other problems during installation, refer to Chapter 9, "Diagnosing and Solving Problems," of the *Microsoft MS-DOS User's Guide.*

Having installed DOS 6, you should consider running DBLSPACE to increase your disk storage capacity and MEMMAKER (if you own a 386+ system) to optimize memory usage. Refer to Chapters 4 and 5, respectively, for further details on these programs.

**NOTE** SETUP moves the files from the old version of DOS into a directory called OLD_DOS.1. If the directory already exists, SETUP will create a new OLD_DOS directory using the lowest available extension number — OLD_DOS.2 or OLD_DOS.3. After you have been using DOS 6 for a few days without any problems, use DELOLDOS to erase all the OLD_DOS directories. After running DELOLDOS, you will not be able to UNINSTAL DOS 6.

## Installing to Floppies

To create a bootable DOS 6 floppy without installing DOS 6 on the hard disk, use the /F switch as follows:

```
a:setup /f
```

Have ready a disk that is the same size as the DOS 6 disks (3 ½-inch or 5 ¼-inch) and label it *Start-up.* The disk does not need to be formatted. Be prepared for a lot of floppy swapping as SETUP transfers files from the DOS 6 disks to the start-up disk. When the installation is complete, SETUP will reboot the system.

Gone are the days when DOS would fit on a single disk. Because of space limitations, SETUP installs the following files on the start-up disk:

IO.SYS
MSDOS.SYS
COMMAND.COM
DBLSPACE.BIN
ATTRIB.EXE
DEBUG.EXE
EXPAND.EXE
FDISK.EXE
FORMAT.COM
RESTORE.EXE

SYS.COM
CHKDSK.EXE
EDIT.COM
QBASIC.EXE
XCOPY.EXE
MSD.EXE
MSAV.EXE
DBLSPACE.EXE

## Installing a Minimum Configuration

If you want to upgrade to DOS 6 but only want to install the minimum number of DOS 6 files needed to upgrade (because you are short on disk space, for example), use the /M switch as follows:

```
a:setup /m
```

SETUP will simply transfer the DOS 6 start-up files to the root directory of the boot drive. You will, however, still need to supply an Uninstall disk, which will be used to store the old DOS start-up files.

After the system is rebooted, you may be surprised to find that you no longer have the CONFIG.SYS and AUTOEXEC.BAT files. I sure was. Presumably, SETUP removes these files because there is a remote chance that some of the commands in these start-up files may conflict with the new version of DOS.

Copies of the original CONFIG.SYS and AUTOEXEC.BAT files are stored on the Uninstall disk with a DAT extension. Copies of these files can also be found in the root directory of the boot drive, with the extension NEW. SETUP may have made modifications to the NEW files, however, such as adding a call to SMARTDRV.EXE in AUTOEXEC.NEW.

Most of the DOS 6 files are stored in compressed form on the installation disks. All the compressed files have an underscore (_) as the third character in the filename extension. You cannot simply copy these files to the hard disk and use them. You must selectively install individual files using the EXPAND program. Copy the EXPAND.EXE file from disk 1 of DOS 6 to a directory on the path. The full syntax for EXPAND is detailed in Part IV, "The Complete DOS 6 Command Reference." But, as an example, the following command copies three files to the DOS subdirectory:

```
expand a:cga.vi_ a:cga.in_ a:cga.gr_ c:\dos
```

View the file PACKING.LST on disk 1 of DOS 6 to see which DOS 6 disk contains the files you want to transfer to the hard disk.

## Installing the Windows Components of DOS 6

If you install DOS 6 and specify that you do not want to install the Windows utilities, you can install them later by running SETUP with the /E switch as follows:

```
a:setup /e
```

SETUP will transfer the Windows utilities MWBACKUP, MWUNDEL, and MWAV, which occupy just less than 3MB of disk space, to the hard drive. SETUP will also add a WINTOOLS group to Program Manager and add a Tools menu to File Manager.

**GEM** You can also use the /E switch with SETUP to install the DOS versions of Backup, Anti-Virus, and Undelete, if you elected not to install them when you first upgraded to DOS 6.

## Installing DOS 6 Manually

If you want to transfer all the DOS 6 files to your hard drive but do not want your system files upgraded, use the /Q switch with SETUP as follows:

```
a:setup /q
```

In most cases, you use the /Q switch only after previously installing the DOS 6 start-up files using /M (the minimum configuration switch discussed earlier).

**NOTE** The Windows files are not installed when SETUP is used with the /M switch. The DOS files consume approximately 4.2MB of disk space.

## Installing to a Nonboot Drive

SETUP provides no convenient way for you to install the external DOS files on a drive other than the boot drive. By running SETUP multiple times using different switches, however, you can get the desired results. The following steps outline the procedure:

1. Execute SETUP with the /M switch to perform a minimal install, which only updates the start-up files.

2. Resurrect your AUTOEXEC.BAT and CONFIG.SYS files using the NEW files in the root directory or the DAT files on the Uninstall disk.
3. Execute SETUP with the /Q switch to perform a manual installation and, when prompted, specify the appropriate drive and directory where SETUP will place the DOS 6 files.
4. Execute SETUP with the /E switch to install the Windows utilities.

## Uninstalling DOS 6

If you have not run DBLSPACE or DELOLDOS since installing DOS 6, you can revert back to your previous operating system by using the Uninstall disk that was used during the DOS 6 installation. Just insert the Uninstall disk into the drive and execute the following command:

```
a:uninstal
```

# Appendix B

# DOS Command Quick Reference

---

This appendix provides a summary of all the commands, device drivers, and ancillary files provided with DOS 6. The programs and files are listed in alphabetical order within the following categories:

- Standard DOS 6 files
- Files installed only when an older version of the file is present on the system
- Files used by SETUP but not installed
- Old files available on the DOS 6 Supplemental disk

## Standard DOS 6 Files

| | |
|---|---|
| ANSI.SYS | Provides ways to spruce up the DOS display capabilities, control the cursor location, and assign special functions to keys. |
| APPEND.EXE | Tells DOS which drives and directories to search in for data files (such as spreadsheets and word processing documents) when a program doesn't find them in the default directory. |
| APPNOTES.TXT | Contains information about running specific programs with DOS 6. |
| ATTRIB.EXE | Sets or clears attributes for one or more files. You can also use ATTRIB to view file attributes. |
| BREAK | Controls how frequently DOS checks to see whether you have pressed Ctrl-Break or Ctrl-C while a command or program is executing. |

| | |
|---|---|
| BUFFERS | Controls the number of disk buffers used by DOS when reading files. *Buffers* are memory blocks designed to store data during disk read and write operations and to improve system performance. |
| CALL | Enables one batch file program to execute a second batch file, after which control is returned to the first batch file. |
| CD | Changes the default (current) directory. |
| CHCP | Used with international character sets to display or change the active code page. |
| CHKDSK.EXE | Checks the integrity of a disk's file structure, reports disk errors, and, optionally, corrects them. CHKDSK displays a summary of the disk's use and the system's memory status. |
| CHOICE.COM | Enables a batch file to prompt the user to select an option by pressing a key. CHOICE sets the ERRORLEVEL to indicate which key was pressed. |
| CLS | Removes all the displayed characters, sets the color to gray on black, and repositions the cursor to the top left corner of the display. |
| COMMAND | Starts a new copy (or instance) of COMMAND.COM, the command interpreter. |
| COMMAND.COM | Displays the command prompt, interprets typed commands, and contains internal MS-DOS commands. |
| COPY | Copies one or more files to another location or to the same location with a new filename. You can instruct COPY, in its extended form, to treat the files as ASCII or binary. |
| COUNTRY.SYS | Configures the country language conventions for your system. |
| CTTY | Enables a different device to be used for keyboard and display I/O. |
| DATE | Sets the system date. |
| DBLSPACE.BIN | Compression kernel — handles all file compression and decompression. |
| DBLSPACE.EXE | Creates and manages compressed drives on hard disks and floppies. |
| DBLSPACE.SYS | Relocates the compression kernel (DBLSPACE.BIN) to low conventional memory or to upper memory. |

| | |
|---|---|
| DEBUG.EXE | Creates and analyzes program files. |
| DEFRAG.EXE | Increases disk performance and enhances data recovery chances by defragmenting files. |
| DEL | Deletes one or more files. |
| DELOLDOS.EXE | Deletes all OLD_DOS directories from a disk. |
| DELTREE.EXE | Removes a directory, all its subdirectories, and all the files in the directories — just like that. |
| DEVICE | Loads a device driver into conventional or base memory. |
| DEVICEHIGH | Loads device drivers into upper memory if it's available or base memory if there is no room in upper memory. |
| DIR | Displays information about one or more files, including the name, size, and date and time of last creation. |
| DISKCOMP.COM | Compares two floppy disks to see whether they contain identical data. |
| DISKCOPY.COM | Duplicates all the data from one floppy disk to another. |
| DISPLAY.SYS | Provides international character support by allowing code-page switching. |
| DOS= | Controls whether DOS is loaded in high memory (on 286 systems and higher) and whether DOS should manage the upper memory area (on 386 systems and higher). |
| DOSHELP.EXE | See FASTHELP. |
| DOSKEY.COM | Provides three primary and very useful functions: the ability to recall previous commands to save from retyping, command-line editing (no more backspace), and a simple but effective macro utility. |
| DOSSHELL.COM | Provides menuing, file-management, and task-switching tools. |
| DRIVER.SYS | Creates logical floppy drives with different characteristics from the physical drive. |
| DRIVPARM | Defines or redefines the properties of disk and tape drives (block devices). |
| ECHO | Controls how batch file commands are displayed during execution. |
| EDIT.COM | Starts the full-screen editor for viewing and editing ASCII (text) files. |
| EGA.CPI | Contains code-page information for EGA monitors. |

| | |
|---|---|
| EGA.SYS | Supports DOSSHELL task swapping on EGA systems. |
| EMM386.EXE | Provides advanced memory-management capabilities on 386 systems and higher. |
| ERASE | See DEL. |
| EXIT | Terminates the current COMMAND.COM session and returns to the preceding one. |
| EXPAND.EXE | Transfers and decompresses files from the DOS distribution disks to your system. |
| FASTHELP.EXE | Provides terse help on nearly all DOS commands but not on device drivers. You can also use DOSHELP. |
| FASTOPEN.EXE | Improves file performance by speeding access to frequently used files. |
| FC.EXE | Compares two files and highlights any differences. |
| FCBS | Specifies the maximum number of file control blocks DOS can have open at any time. |
| FDISK.EXE | Prepares a new hard disk for formatting and manages logical drive assignments on a partitioned disk. |
| FILES | Specifies the number of files that DOS can have open concurrently. |
| FIND.EXE | Searches for specific text in a file or files. |
| FOR | Executes a command or statement multiple times and applies a new string or filename with each iteration. |
| FORMAT.COM | Formats a floppy or hard disk and creates a new file allocation table; optionally marks bad sectors. |
| GOTO | Instructs DOS to continue processing batch file commands at a specific location in the batch file. |
| GRAPHICS.COM | Enables you to print graphics screens just by pressing PrtScr on CGA, EGA, and VGA systems. |
| HELP.COM | Displays full-screen on-line help for DOS commands and device drivers. |
| HIMEM.SYS | Manages extended memory on 286 systems and higher. |
| IF | Provides a conditional branch in a batch file execution. |
| INCLUDE | Instructs DOS to process a specific block of commands when used in a multiboot CONFIG.SYS file. |
| INTERLNK.EXE | Changes the drive assignments for two computers connected by way of InterLink. |

| | |
|---|---|
| INTERSVR.EXE | Parallel or serial file-transfer server. |
| IO.SYS | The DOS basic input/output system (BIOS) that prepares the system at start-up and contains resident device drivers. |
| KEYB.COM | Configures the keyboard for international (non-USA) use. |
| KEYBOARD.SYS | A device driver that supports different keyboard layouts. |
| LABEL.EXE | Creates, changes, or deletes a disk volume label. |
| LASTDRIVE | Specifies how many logical drives DOS recognizes. |
| LH | See LOADHIGH. |
| LOADFIX.COM | Fills the first 64K of conventional memory so that no programs load there. |
| LOADHIGH | Loads programs in upper memory. |
| MD | Creates a new directory. |
| MEM.EXE | Displays valuable information that shows how your system's memory is utilized and how much free memory is available. |
| MEMMAKER.EXE | Assesses your memory configuration and modifies CONFIG.SYS and AUTOEXEC.BAT to make optimum use of memory (for 386 and higher systems). |
| MENUCOLOR | Configures the display colors for a multiconfiguration menu. |
| MENUDEFAULT | Sets the default item in a multiconfiguration menu and optionally the number of seconds to wait for user input before applying the default. |
| MENUITEM | Identifies a menu option in a multiconfiguration menu. |
| MODE.COM | Configures devices such as printers, monitors, keyboards, and communication ports. |
| MORE.COM | Pauses the output from a command-line utility after every screen. |
| MOVE.EXE | Moves files from one path to another and renames directories. |
| MSAV.EXE | Runs the full-screen utility Microsoft Anti-Virus, which scans drives for viruses and optionally removes them. Installed when non-Windows tools are selected from SETUP. |
| MSBACKUP.EXE | Runs the full-screen utility Microsoft Backup, which backs up and restores copies of your files. Installed when non-Windows tools are selected from SETUP. |

| | |
|---|---|
| MSCDEX.EXE | Makes a CD-ROM behave like a standard disk device and assigns a logical drive letter to the CD-ROM drive. |
| MSD.EXE | Analyzes and reports on your detailed system configuration. |
| MSDOS.SYS | The DOS kernel — manages files, directories, memory, and environments. |
| MSTOOLS.DLL | Windows 3.*x* File Manager extensions. Installed when Windows tools are selected from SETUP. |
| MWAV.EXE | Windows program that scans drives for viruses and optionally removes them. Installed when Windows tools are selected from SETUP. |
| MWAVTSR.EXE | Provides Windows support for messages generated by VSAFE. |
| MWBACKUP.EXE | Backup for Windows. Installed when Windows tools are selected from SETUP. |
| MWUNDEL.EXE | Windows version of UNDELETE to recover accidentally deleted files and directories. Installed when Windows tools are selected from SETUP. |
| NLSFUNC.EXE | Provides *N*ational *L*anguage *S*upport by enabling code-page switching. |
| NUMLOCK | Controls whether the keyboard Num Lock key is set to On or Off at system boot. |
| PACKING.LST | Lists the files located on each DOS 6 disk. |
| PATH | Stores a list of directories that DOS searches when it looks for a program or batch file. |
| PAUSE | Temporarily halts batch file execution and waits for the user to press a key. |
| POWER.EXE | Reduces power consumption to prolong battery life on systems compliant with APM (Advanced Power Management). |
| PRINT.EXE | Provides background printing capabilities that enable you to proceed with tasks while one or more files are being printed. |
| PROMPT | Customizes the prompt DOS displays when it is ready to receive a command. |
| QBASIC.EXE | Invokes the full-screen development environment for creating and running BASIC language programs. |
| RAMDRIVE.SYS | Creates a RAM disk (a virtual disk in memory) to improve performance. |

| | |
|---|---|
| RD | Removes (that is, deletes) an empty directory. |
| README.TXT | Contains information that is hardware specific or was available too late to be printed in the manuals. |
| REM | Provides a way to add comments to any batch file and CONFIG.SYS. |
| REN | See RENAME. |
| RENAME | Changes the name of one or more files. |
| REPLACE.EXE | Synchronizes the contents of two directories. |
| RESTORE.EXE | Restores files that were stored on floppy disks by using the original DOS BACKUP utility. |
| RMDIR | See RD. |
| SETUP.EXE | Runs the DOS 6 installation and setup program. |
| SETVER.EXE | Displays and modifies the version table in which programs and fake version numbers are stored. This enables programs that expect a specific version of DOS to be installed to run with DOS 6. |
| SHARE.EXE | Allows file sharing and locking for multiuser or multiapplication access. |
| SHELL | Sets the options for COMMAND.COM or enables a different command processor to be loaded. |
| SHIFT | Used in batch files to drop the first passed parameter (%1) and shift all other parameters left one position. |
| SIZER.EXE | Used by MEMMAKER during the optimization process; it is not intended for general use. |
| SMARTDRV.EXE | Creates a disk cache in expanded or extended memory. |
| SMARTMON.EXE | Monitors SMARTDRV cache performance under Windows. |
| SORT.EXE | Sorts input data and directs the sorted output to a device or file. |
| STACKS | Allocates memory to manage hardware interrupts. |
| SUBMENU | Provides a way to branch to a secondary menu in a multiconfiguration system. |
| SUBST.EXE | Assigns a drive letter alias to a specific directory. |
| SWITCHES | A miscellaneous collection of four configuration switches for customizing the behavior of the operating system. |
| SYS.COM | Makes a formatted drive bootable or updates the DOS version by transferring the primary DOS files IO.SYS, MSDOS.SYS, COMMAND.COM, and DBLSPACE.BIN to the root directory of a drive. |

| | |
|---|---|
| TIME | Displays and sets the system time. |
| TREE.COM | Displays the directory structure of an entire drive or for a specific directory and its subdirectories. |
| TYPE | Displays the contents of an ASCII file. |
| UNDELETE.EXE | Provides mechanisms for enhancing the chances of recovering deleted files and tries to undelete files that have been deleted. |
| UNFORMAT.COM | Recovers files and directories destroyed by an accidental FORMAT. |
| UNINSTAL.EXE | Restores the previous version of DOS after installing the DOS 6 Upgrade. (Created on the Uninstall disk.) |
| VER | Displays the DOS version number. |
| VERIFY | Instructs DOS to verify that data just written to disk can be read back into memory. |
| VOL | Displays a disk's volume label and serial number. |
| VSAFE.COM | Monitors disk activity and protects against virus infection. |
| WINA20.386 | Enables DOS 6 to run with Microsoft Windows 3.0 in 386 Enhanced Mode. |
| WNTOOLS.GRP | Windows Tools program group. Installed when Windows tools are selected from SETUP. |
| XCOPY.EXE | An enhanced version of COPY that can copy files and directories and that loads multiple files in memory to improve performance. |

## Files Installed Only When an Older Version of the File Is Present on the System

| | |
|---|---|
| DMDRVR.BIN | A device driver for the Disk Manager disk partitioning program |
| MOUSE.COM | Microsoft Mouse driver |
| MSCDEX.EXE | Microsoft CD-ROM extensions |
| SSTOR.SYS | A device driver for the SpeedStor disk partitioning program |

## Files Used by SETUP but Not Installed

| | |
|---|---|
| SETUP.EXE | DOS 6 Upgrade installation program |
| AV.GRP | Renamed as WNTOOLS.GRP during setup |
| CONFIG.SYS | Base configuration |
| AUTOEXEC.BAT | Base configuration |

## Old Files Available on the DOS 6 Supplemental Disk

| | |
|---|---|
| 4201.CPI | Code-page information for IBM Proprinters II and III, Models 4201 and 4202 |
| 4208.CPI | Code-page information for IBM Proprinters X24E Model 4207, XL24E Model 4208, and compatibles |
| 5202.CPI | Code-page information for IBM Quietwriter III Printer Model 5202 and compatibles |
| ASSIGN.COM | Redirects requests for disk operations on one drive to a different drive |
| BACKUP.EXE | The old MS-DOS backup program; replaced by MSBACKUP.EXE |
| COMP.EXE | Compares the contents of two files or two sets of files byte by byte |
| CV.COM | Update for older versions of the CodeView debugging program |
| EDLIN.EXE | A line-oriented text editor |
| EXE2BIN.EXE | Converts executable files to binary format |
| GORILLA.BAS | A sample QBasic program |
| GRAFTABL.COM | Enables the display of extended characters of a specified code page in Graphics Mode |
| JOIN.EXE | Joins a disk drive with a directory on another disk drive |
| LCD.CPI | Contains code-page information for certain liquid crystal displays |
| MIRROR.COM | Tracks deleted file information |

| | |
|---|---|
| MONEY.BAS | A sample QBasic program |
| MSHERC.COM | Provides support for a Hercules monitor running with EDIT or QBASIC |
| NIBBLES.BAS | A sample QBasic program |
| PRINTER.SYS | Supports code-page switching for parallel port printers |
| PRINTFIX.COM | Disables verification of a printer's status line |
| RECOVER.EXE | Recovers readable information from a defective disk |
| REMLINE.BAS | A sample QBasic program |

# Appendix C

# ANSI.SYS Quick Reference

This appendix provides a summary of all the ANSI.SYS commands for controlling the display, controlling cursor movement, and reassigning keys.

All ANSI.SYS commands are prefixed by two special characters: the Esc character, which has the ASCII value 027, and the left square-bracket character ([), which has the ASCII value 019. In EDIT, you can enter the Esc code by pressing Ctrl-P and then pressing the Esc key. EDIT displays the Esc character as a left-pointing arrow (←).

The ANSI.SYS commands are summarized in the following categories:

- Color display codes used with `Esc[`*code*`m`
- Display attribute codes used with `Esc[`*code*`m`
- Text erasing codes
- Cursor-control codes
- Display mode control codes
- Keyboard control codes

## Color Display Codes Used with `Esc[`*code*`m`

| *Color* | *Foreground Code* | *Background Code* |
|---|---|---|
| Black | 30 | 40 |
| Red | 31 | 41 |
| Green | 32 | 42 |
| Brown | 33 | 43 |
| Blue | 34 | 44 |

| *Color* | *Foreground Code* | *Background Code (continued)* |
|---|---|---|
| Magenta | 35 | 45 |
| Cyan | 36 | 46 |
| Light gray | 37 | 47 |

## Display Attribute Codes Used with `Esc[`***code***`m`

| *Code* | *Attribute* |
|---|---|
| 0 | All attributes off |
| 1 | Bright intensity |
| 4 | Underline (monochrome systems only) |
| 5 | Blinking text |
| 7 | Reverse video |
| 8 | Hidden or invisible |

## Text Erasing Codes

| *Code* | *Description* |
|---|---|
| `Esc[2J` | Clears the screen |
| `Esc[K` | Erases line from cursor position to end of line |

## Cursor-Control Codes

| *Escape Sequence* | *Action* |
|---|---|
| `Esc[`*row*`;`*col*`H` | Moves the cursor to the specific row and column. If only one value is specified (without a semicolon), it is assumed to represent the column, and the row defaults to 1. |
| `Esc[s` | Saves the active cursor position. |
| `Esc[u` | Repositions the cursor to the position it occupied when the last s (save) command was issued. |
| `Esc[`*pos*`A` | Moves the cursor up the number of rows specified by the *pos* variable. The cursor stops when it reaches the first row. |

*(continued)*

| *Escape Sequence* | *Action (continued)* |
|---|---|
| `Esc[`*pos*`B` | Moves the cursor down *pos* rows. The cursor stops when it reaches the last row. |
| `Esc[`*pos*`C` | Moves the cursor to the right *pos* columns. The cursor stops when it reaches the last column. |
| `Esc[`*pos*`D` | Moves the cursor left *pos* columns. The cursor stops when it reaches the first column. |

## Display Mode Control Codes

| *Escape Sequence* | *Description* |
|---|---|
| `Esc[=0h` | Text — monochrome, 40 columns by 25 lines |
| `Esc[=1h` | Text — color, 40 columns by 25 lines |
| `Esc[=2h` | Text — monochrome, 80 columns by 25 lines |
| `Esc[=3h` | Text — color, 80 columns by 25 lines |
| `Esc[=4h` | Graphics — 4-color, 320 by 200 pixels |
| `Esc[=5h` | Graphics — monochrome, 320 by 200 pixels |
| `Esc[=6h` | Graphics — monochrome, 640 by 200 pixels |
| `Esc[=7h` | Enables line wrapping |
| `Esc[=13h` | Graphics — color, 320 by 200 pixels |
| `Esc[=14h` | Graphics — 16-color, 640 by 200 pixels |
| `Esc[=15h` | Graphics — monochrome, 640 by 350 pixels |
| `Esc[=16h` | Graphics — 16-color, 640 by 350 pixels |
| `Esc[=17h` | Graphics — monochrome, 640 by 480 pixels |
| `Esc[=18h` | Graphics — 16-color, 640 by 480 pixels |
| `Esc[=19h` | Graphics — 256-color, 320 by 200 pixels |

## Keyboard Control Codes

| *Key* | *Standard* | *With Shift* | *With Ctrl* | *With Alt* |
|---|---|---|---|---|
| A | 97 | 65 | 1 | 0;30 |
| B | 98 | 66 | 2 | 0;48 |
| C | 99 | 67 | 3 | 0;46 |
| D | 100 | 68 | 4 | 0;32 |
| E | 101 | 69 | 5 | 0;18 |
| F | 102 | 70 | 6 | 0;33 |
| G | 103 | 71 | 7 | 0;34 |
| H | 104 | 72 | 8 | 0;35 |
| I | 105 | 73 | 9 | 0;23 |
| J | 106 | 74 | 10 | 0;36 |
| K | 107 | 75 | 11 | 0;37 |
| L | 108 | 76 | 12 | 0;38 |
| M | 109 | 77 | 13 | 0;50 |
| N | 110 | 78 | 14 | 0;49 |
| O | 111 | 79 | 15 | 0;24 |
| P | 112 | 80 | 16 | 0;25 |
| Q | 113 | 81 | 17 | 0;16 |
| R | 114 | 82 | 18 | 0;19 |
| S | 115 | 83 | 19 | 0;31 |
| T | 116 | 84 | 20 | 0;20 |
| U | 117 | 85 | 21 | 0;22 |
| V | 118 | 86 | 22 | 0;47 |
| W | 119 | 87 | 23 | 0;17 |
| X | 120 | 88 | 24 | 0;45 |
| Y | 121 | 89 | 25 | 0;21 |
| Z | 122 | 90 | 26 | 0;44 |
| 1 | 49 | 33 | N/A | 0;120 |
| 2 | 50 | 64 | 0 | 0;121 |
| 3 | 51 | 35 | N/A | 0;122 |
| 4 | 52 | 36 | N/A | 0;123 |
| 5 | 53 | 37 | N/A | 0;124 |
| 6 | 54 | 94 | 30 | 0;125 |

*(continued)*

| *Key* | *Standard* | *With Shift* | *With Ctrl* | *With Alt (continued)* |
|---|---|---|---|---|
| 7 | 55 | 38 | N/A | 0;126 |
| 8 | 56 | 42 | N/A | 0;126 |
| 9 | 57 | 40 | N/A | 0;127 |
| 0 | 48 | 41 | N/A | 0;129 |
| - | 45 | 95 | 31 | 0;130 |
| = | 61 | 43 | N/A | 0;131 |
| [ | 91 | 123 | 27 | 0;26 |
| ] | 93 | 125 | 29 | 0;27 |
| Spacebar | 92 | 124 | 28 | 0;43 |
| ; | 59 | 58 | N/A | 0;39 |
| ' | 39 | 34 | N/A | 0;40 |
| , | 44 | 60 | N/A | 0;51 |
| . | 46 | 62 | N/A | 0;52 |
| / | 47 | 63 | N/A | 0;53 |
| ` | 96 | 126 | N/A | 0;41 |
| Enter (keypad) | 13 | N/A | 10 | 0;166 |
| / (keypad) | 47 | 47 | 0;142 | 0;74 |
| * (keypad) | 42 | 0;144 | 0;78 | N/A |
| - (keypad) | 45 | 45 | 0;149 | 0;164 |
| + (keypad) | 43 | 43 | 0;150 | 0;55 |
| 5 (keypad) | 0;76 | 53 | 0;143 | N/A |
| F1 | 0;59 | 0;84 | 0;94 | 0;104 |
| F2 | 0;60 | 0;85 | 0;95 | 0;105 |
| F3 | 0;61 | 0;86 | 0;96 | 0;106 |
| F4 | 0;62 | 0;87 | 0;97 | 0;107 |
| F5 | 0;63 | 0;88 | 0;98 | 0;108 |
| F6 | 0;64 | 0;89 | 0;99 | 0;109 |
| F7 | 0;65 | 0;90 | 0;100 | 0;110 |
| F8 | 0;66 | 0;91 | 0;101 | 0;111 |
| F9 | 0;67 | 0;92 | 0;102 | 0;112 |
| F10 | 0;68 | 0;93 | 0;103 | 0;113 |
| F11 | 0;133 | 0;135 | 0;137 | 0;139 |
| F12 | 0;134 | 0;136 | 0;138 | 0;140 |

*(continued)*

| *Key* | *Standard* | *With Shift* | *With Ctrl* | *With Alt (continued)* |
|---|---|---|---|---|
| Home | 0;71 | 55 | 0;119 | N/A |
| Up arrow | 0;72 | 56 | 0;141 | N/A |
| PgUp | 0;73 | 57 | 0;132 | N/A |
| Left arrow | 0;75 | 52 | 0;115 | N/A |
| Right arrow | 0;77 | 54 | 0;116 | N/A |
| End | 0;79 | 49 | 0;117 | N/A |
| Down arrow | 0;80 | 50 | 0;145 | N/A |
| PgDn | 0;81 | 51 | 0;118 | N/A |
| Insert | 0;82 | 48 | 0;146 | N/A |
| Delete | 0;83 | 46 | 0;147 | N/A |
| Print Screen | N/A | N/A | 0;114 | N/A |
| Pause | N/A | N/A | 0;0 | N/A |
| Backspace | 8 | 8 | 127 | 0 |
| Enter | 13 | N/A | 10 | 0 |
| Tab | 9 | 0;5 | 0;148 | 0;165 |
| Home (gray key) | 224;71 | 224;71 | 224;119 | 224;151 |
| Up arrow (gray key) | 224;72 | 224;72 | 224;141 | 224;152 |
| PgUp (gray key) | 224;73 | 224;73 | 224;132 | 224;153 |
| Left arrow (gray key) | 224;75 | 224;75 | 224;115 | 224;155 |
| Right arrow (gray key) | 224;77 | 224;77 | 224;116 | 224;157 |
| End (gray key) | 224;79 | 224;79 | 224;117 | 224;159 |
| Down arrow (gray key) | 224;80 | 224;80 | 224;145 | 224;154 |
| PgDn (gray key) | 224;81 | 224;81 | 224;118 | 224;161 |
| Ins (gray key) | 224;82 | 224;82 | 224;146 | 224;162 |
| Del (gray key) | 224;83 | 224;83 | 224;147 | 224;163 |

# Appendix D

# Interacting with Workgroups and Networks

## The InterLink Connection

If you just want to connect two PCs together occasionally to transfer files or share a printer, you don't need to invest in a network. DOS 6 offers InterLink commands for providing these basic services. InterLink is primarily designed to connect a laptop PC to a desktop PC, but you can use it to connect any two PCs that can be wired together using serial or parallel ports.

### Wiring the two computers together

You can use three different types of cable to connect two computers: a bi-directional parallel cable, a three-wire serial cable, or a seven-wire null modem serial cable. When choosing cables, consider the following pros and cons:

- Because parallel cables use several lines simultaneously, parallel connections are inherently faster than serial connections. The cable, however, is bulkier and must be connected to the parallel printer port — that is, a female DB25 connector. Many older desktop systems have only one parallel port, and if the port is being used for InterLink, it cannot also be connected to a printer.
- File transfer speeds are significantly slower when you use serial connections because data is transferred along only one "wire." Most PCs have a spare serial port. If no spare is available, you usually can temporarily disconnect an external modem or mouse to free a serial port. If you use a serial connection, the InterLink software can be remotely installed on one PC.
- Each computer can use a different port number, but both computers must use the same port type. For example, you can connect a serial cable to COM1 on one computer and COM3 on the other.

Because I don't perform large file transfers very often, I use the three-wire serial cable option. Three-wire serial cable comprises a length of standard telephone cable with a nine-pin (DB9) serial connection at each end. It's not the fastest configuration, but the cable is thin enough to carry around in my notebook PC carrying case.

## Establishing an InterLink connection

The InterLink tools are provided in the form of two programs: INTERLNK.EXE and INTERSVR.EXE. When you use InterLink to connect two computers, you run INTERLNK on one computer and INTERSVR on the other.

NOTE

MS-DOS documentation refers to the PC running INTERSVR as the *server* and to the computer running INTERLNK as the *client.*

### How to install INTERLNK

INTERLNK.EXE is both a device driver and an executable program. Install INTERLNK.EXE in CONFIG.SYS in the computer that will be fully functional and will have access to the drives and printers attached to the remote computer. By default, INTERLNK.EXE automatically loads high into available UMBs, so you do not need to load it using DEVICEHIGH.

INTERLNK.EXE supports a variety of switches, all of which are documented in Part IV, "The Complete DOS 6 Command Reference." One frequently used switch is /DRIVES. By default, INTERLNK.EXE only provides access to three drives on the remote (server) computer. Use the /DRIVES switch to access drives beyond drive C. The following statement in CONFIG.SYS provides access to drives A through E on the remote printer:

```
device=c:\dos\interlnk.exe /drives:5
```

NOTE

You may also need to add or modify the LASTDRIVE= statement in the CONFIG.SYS file to ensure that the client system can accommodate these additional drive letters.

You can use a variety of switches to reduce the memory requirements of INTERLNK.EXE. By default, INTERLNK.EXE scans the installed serial and parallel ports, looking for a connection to a server computer. To save memory, you can instruct the device driver to use an explicit port instead of scanning for a connection. Use the /LPT*n* or /COM*n* switch, replacing *n* with the actual port number — for example, /LPT1. Also, if you do not plan to use the server's printer, specify /NOPRINTER to save memory.

Install INTERLNK.EXE only when you intend to connect one computer to another. Don't waste valuable memory on the device driver if you don't plan to use it. Use the /AUTO switch to instruct INTERLNK.EXE to install only if it can establish an immediate connection with a server. If a server is not found, the device driver is not loaded, and no memory is wasted. If you use the /AUTO switch, make sure that the server machine is actively running the INTERSVR.EXE program before you boot the client computer.

The following statement from CONFIG.SYS installs INTERLNK.EXE (when the server is active) via COM2 and provides access to drives A through E on the remote computer. It does not provide printer support.

```
device=c:\dos\interlnk.exe /drives:5 /com2 /auto /noprinter
```

## How to install INTERSVR

The server computer needs no special device drivers. To establish a connection, the server computer simply executes the INTERSVR command. INTERSVR supports a variety of switches (see the Command Reference for details) but is most often executed with no switches, as follows:

```
intersvr
```

By default, INTERSVR redirects all drives to make them available to the client computer. You can use the *drive*: and /X=*drive*: switches to control which drives are available to the client. To explicitly identify which drives should be redirected, use the *drive*: switch and specify one or more individual drives; to exclude certain drives, use the /X=*drive*: switch, specifying one or more drives.

When a successful connection is established, the server computer drives are assigned to additional drive letters on the client computer. The server computer displays the status of the connection and a list of the drive connections. The list shows the server drive letter and printer ports along with the corresponding drive letter and port assignments on the client.

NOTE

If INTERLNK.EXE is not installed on the remote computer, you can transfer the INTERLNK.EXE file by using INTERSVR with the /RCOPY switch (for remote copy).

## Three easy steps to successful linking

To recap, you should take the following steps to connect two computers with InterLink:

1. Wire the two computers using a bidirectional parallel cable, a three-wire serial cable, or a seven-wire null modem cable.

2. Execute the INTERSVR.EXE program on the computer that is to act as the server.
3. Boot the client computer with INTERLNK.EXE installed as a device driver. Remember to use the /DRIVES switch if the server has more than one hard drive.

**NOTE** You can boot the client computer before the server computer starts running INTERSVR if the /AUTO switch was not used with the device driver. After INTERSVR is started, the client computer should execute INTERLNK.EXE to establish the connection.

## Using server resources

To check the drive and port assignments, just enter the INTERLNK command on the client computer. When you issue the command, INTERLNK displays the active settings. The following is an example display:

```
Port=COM1

      This Computer          Other Computer
        (Client)                (Server)
--------------------------------------------------------------
        E:     equals    A:
        F:     equals    B:
        G:     equals    C: (211Mb) UPSTAIRS-C
        H:     equals    D: (173Mb) UPSTAIRS-D
        I:     equals    E: (167Mb) UPSTAIRS-E
        LPT2: equals     LPT1:
```

To access the server drives, use the drive and port aliases listed in the first (client) column. You have full read and write access to the remote drives.

If, for example, the settings are as shown in the sample display, you can edit the AUTOEXEC.BAT file on the server computer by entering the following command:

```
edit g:\autoexec.bat
```

Similarly, you can access the server's printer by outputting data to the LPT2 port.

**TIP** One of my old computers (Bessie) does not have a 3½-inch drive. I can still install software from 3½-inch floppies by using InterLink to link my notebook computer (Ralph) to Bessie. Because InterLink provides access to the floppies on the server, I can install software to my old computer using the notebook's drive A.

## Understanding InterLink limitations

Only local hard disks and standard removable drives (such as floppies and Bernoullies) can be redirected to the client computer. InterLink does not support network drives, CD-ROMs, or devices such as modems, fax cards, and scanners.

Unlike some third-party products, such as LapLink Pro, InterLink does not support a dial-up file transfer facility. In other words, the client cannot dial up the server and establish a connection via a modem. In addition, InterLink is slower than similar third-party products.

When you run INTERSVR from within Windows, DOS Shell, or any multitasker, you cannot switch tasks. You must terminate INTERSVR to return to other tasks.

You cannot execute the following commands on a server drive: CHKDSK, DEFRAG, DISKCOMP, DISKCOPY, FDISK, FORMAT, MIRROR, SYS, UNDELETE, and UNFORMAT.

But, hey, the price is right.

# Workgroup Wizardry

To provide DOS 6 with rudimentary networking capabilities, Microsoft offers a separate product called *Workgroup Connection.* This product allows any DOS 6 computer to connect to a Windows for Workgroups, Windows NT, Microsoft LAN Manager, or compatible network. The remainder of this appendix explains how to use the Workgroup Connection software and includes a supplementary Command Reference detailing the complete syntax for all the Workgroup Connection commands.

To gain access to a network, all your system needs is a network card (or plug-on network adapter) and the appropriate wiring to connect into the network (coaxial, twisted pair, or fiber optic cable).

A DOS 6 system using Workgroup Connection can access disk drives, CD-ROMs, and printers connected to other computers in the network (if these other computers are running full network software), but not tape drives, scanners, and modems. Such a system, however, cannot share its local disk drives and printers with other users. In other words, you can access devices on other computers that are running full network software, but those other systems cannot access your devices. If you need to share your system's local devices with other users, you must install additional network software that is compatible with the network software used by your workgroup — for example, Windows for Workgroups.

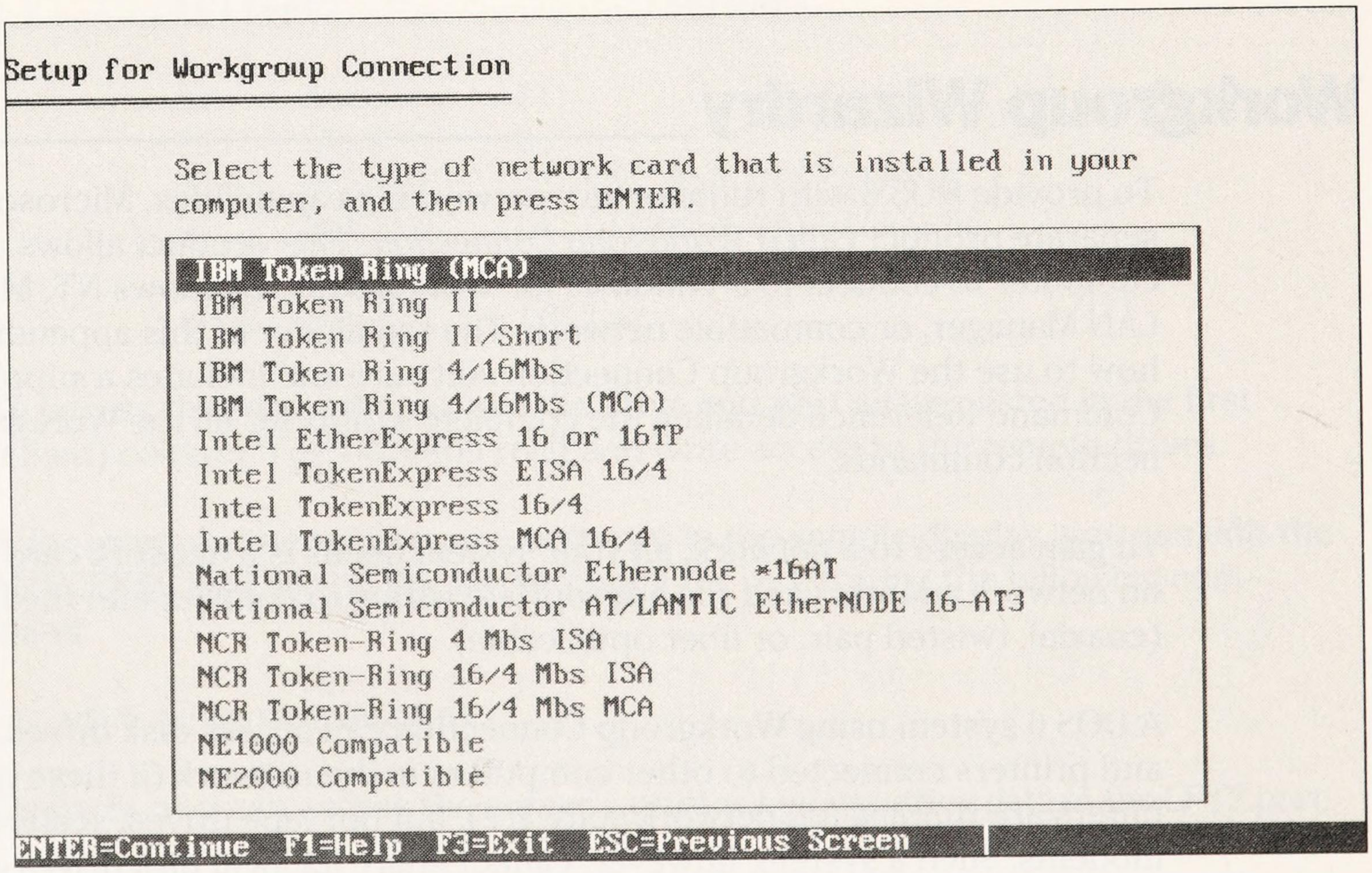
In addition to the basic network software needed to communicate with other *nodes* (systems on the network) and access their devices, DOS 6 includes a copy of Microsoft Mail 3.0. If your workgroup uses Microsoft Mail, you can participate fully in workgroup electronic mail.

## Installing the network software

**CAUTION**

DOS 6 Workgroup Connection files cannot be installed on a Vertisoft System DoubleDisk drive. Also, you should not install the Workgroup Connection on a machine that already has LAN Manager or Windows for Workgroups installed.

Before you install the Workgroup Connection software on your system, install the network card and cabling. The software drivers that you will install are dependent on the specific brand of network card in use, and the setup program will try to detect the type of network card. If the program cannot determine the card type, you are prompted to select one from a list, as shown in Figure D-1.



```
Setup for Workgroup Connection
══════════════════════════════

          Select the type of network card that is installed in your
          computer, and then press ENTER.

          IBM Token Ring (MCA)
           IBM Token Ring II
           IBM Token Ring II/Short
           IBM Token Ring 4/16Mbs
           IBM Token Ring 4/16Mbs (MCA)
           Intel EtherExpress 16 or 16TP
           Intel TokenExpress EISA 16/4
           Intel TokenExpress 16/4
           Intel TokenExpress MCA 16/4
           National Semiconductor Ethernode *16AT
           National Semiconductor AT/LANTIC EtherNODE 16-AT3
           NCR Token-Ring 4 Mbs ISA
           NCR Token-Ring 16/4 Mbs ISA
           NCR Token-Ring 16/4 Mbs MCA
           NE1000 Compatible
           NE2000 Compatible

ENTER=Continue  F1=Help  F3=Exit  ESC=Previous Screen
```

**Figure D-1:** SETUP displays a list of supported network cards.

On lower power systems, such as 8086-based PCs and XTs, SETUP can take as long as five minutes to check system configurations.

During the installation procedure, you are prompted to supply the name of the workgroup you want to join. If you do not know the workgroup name, ask the network administrator or a colleague for the correct name. Don't just make up a name; if you do, the installation will not be successful. A colleague using Windows for Workgroups or Windows NT can verify the workgroup name by selecting the Connect Network Drive option from the Disk menu in File Manager.

In addition to the workgroup name, you need to specify two additional unique names: your user name and your computer name. In most (organized) workgroups, user names are assigned according to some standard, such as the first five characters of the user's last name plus the first two initials. The computer name often reflects the location of the computer; for example, the room number might be used as the computer name. Check with the network administrator to determine appropriate user and computer names.

NOTE

The workgroup name, computer name, and user name can each be up to 15 characters long. Names can include letters, numbers, or the following characters:

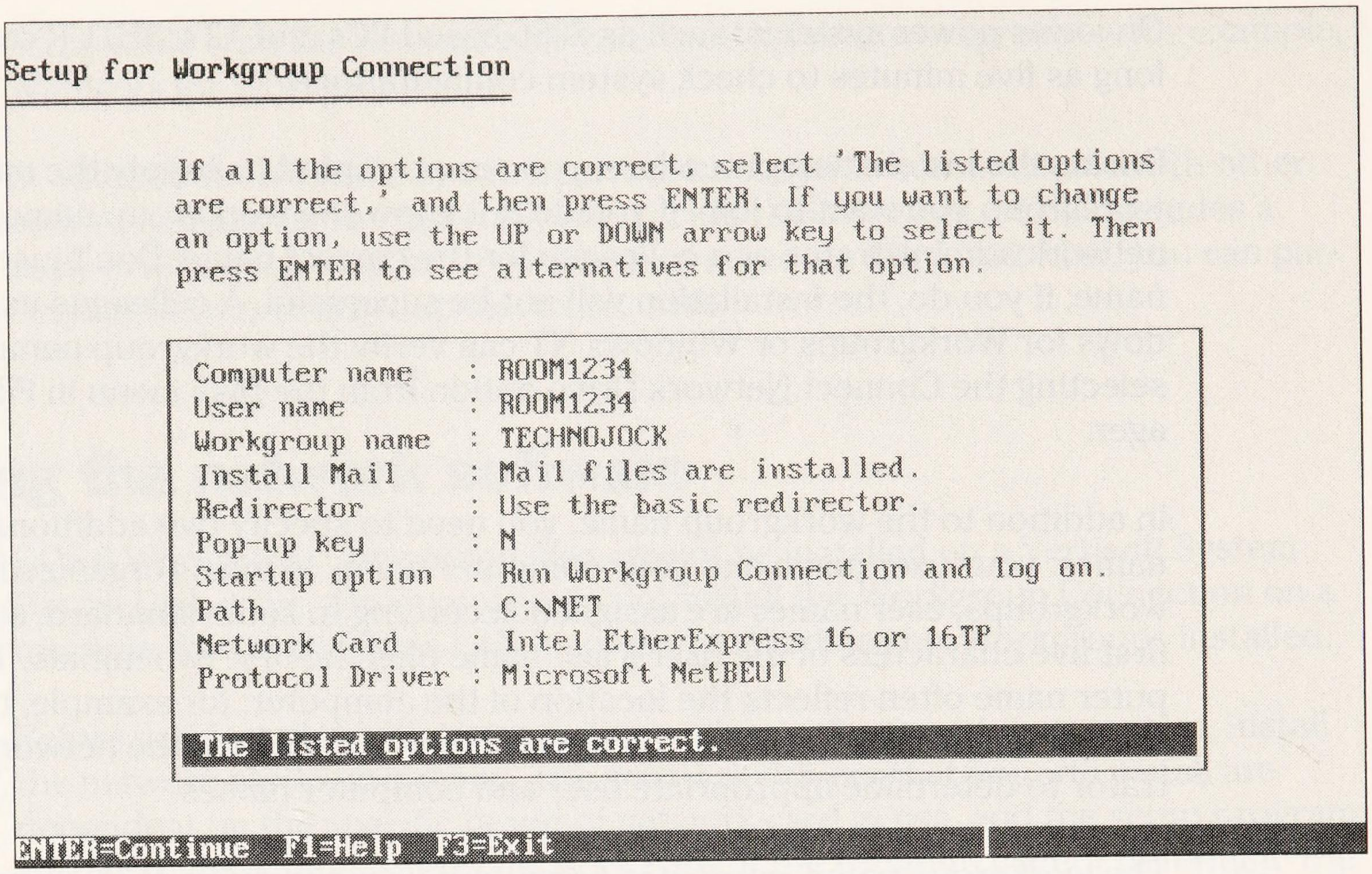
! # $ % & ( ) ^ _ ‘ { } and ~

Sometimes, the first eight characters of the user name are used by a network program to create files; use a naming convention that ensures that the first eight characters of each user name are unique.

Armed with this information, you can install the Workgroup Connection software. Insert the Workgroup Connection setup disk and execute the SETUP program from it.

If SETUP aborts, displaying a message indicating that it cannot update or edit the file PROTOCOL.INI, you may have insufficient memory available. Make more memory available and try again.

Figure D-2 shows the settings screen that SETUP displays after it assesses your system configuration and prompts you for the computer and workgroup names. Notice that the program assumes that the computer name and user name are the same. To change any option, use the cursor keys to highlight the option and press Enter.



```
Setup for Workgroup Connection

    If all the options are correct, select 'The listed options
    are correct,' and then press ENTER. If you want to change
    an option, use the UP or DOWN arrow key to select it. Then
    press ENTER to see alternatives for that option.

      Computer name    : ROOM1234
      User name        : ROOM1234
      Workgroup name   : TECHNOJOCK
      Install Mail     : Mail files are installed.
      Redirector       : Use the basic redirector.
      Pop-up key       : N
      Startup option   : Run Workgroup Connection and log on.
      Path             : C:\NET
      Network Card     : Intel EtherExpress 16 or 16TP
      Protocol Driver  : Microsoft NetBEUI

      The listed options are correct.

ENTER=Continue  F1=Help  F3=Exit
```

**Figure D-2:** SETUP displays the Workgroup Connection settings.

## Reviewing network commands

The main command for managing workgroups is appropriately named NET. The NET command supports a whole family of subcommands, which are summarized in Table D-1.

The Pop-Up program provides an easy way to control network settings by using menus and dialog boxes rather than command switches. Pop-Up is discussed in detail later in this chapter.

## Connecting to a network

During the Workgroup Connection setup, SETUP makes modifications to CONFIG.SYS and AUTOEXEC.BAT. These modifications are to load network drivers and automate the network log-on procedure.

Before modifying CONFIG.SYS and AUTOEXEC.BAT, SETUP makes backup copies, assigning an extension of 001 (or the next available number).

### Table D-1 Net Command Summary

| *Command* | *Description* |
|---|---|
| NET CONFIG | Shows the active workgroup settings, including the computer name, user name, software version, workstation directory, and workgroup. |
| NET LOGOFF | Disconnects the computer from the workgroup and displays all connections to disk drives and printers that are being disconnected. |
| NET LOGON | Connects your workstation to a workgroup. If a password is not specified on the command line, you are prompted to enter it. |
| NET PASSWORD | Changes the log-on password. A password is not case sensitive. |
| NET PRINT | Manages shared printers in the workgroup and the print jobs for your workstations. |
| NET START | Establishes a workgroup connection and optionally loads the Pop-Up utility to simplify creating drive and printer connections. |
| NET STOP | Disconnects a workstation and optionally unloads the Pop-Up TSR. |
| NET TIME | Synchronizes your computer's date and time with the network. This command only works on a network using LAN Manager, which has a designated time server. |
| NET USE | Connects or disconnects network drives and printers. |
| NET VER | Displays the version number of the network redirector. |
| NET VIEW | Provides an easy way to determine the computer names of the other computers in the workgroup, along with names of the shared directories and printers. |

## Network entries in CONFIG.SYS

The following is an example of the network commands added to CONFIG.SYS:

```
device=C:\NET\protman.dos /i:C:\NET
device=C:\NET\workgrp.sys
device=C:\NET\exp16.dos
LASTDRIVE=Z
```

NOTE

In most circumstances, you can load all three device drivers into UMBs using DEVICEHIGH. Use the network for a few days with the device drivers loaded low to ensure that no other network problems exist and then use MEMMAKER to optimize your configuration. If you experience problems with the device drivers loaded high, change the DEVICEHIGH statements back to DEVICE. Then add the three device driver names to MEMMAKER.INF, placing an asterisk before each device driver name, and run MEMMAKER again. (Refer to Chapter 5 for more information on MEMMAKER.)

The PROTMAN.DOS device driver examines the settings in the file PROTOCOL.INI (located in the directory specified with the /I switch) and then binds the appropriate network protocols with the network adapter card device driver (in the example, EXP16.DOS). The PROTMAN.DOS device driver must be located before other network drivers in CONFIG.SYS.

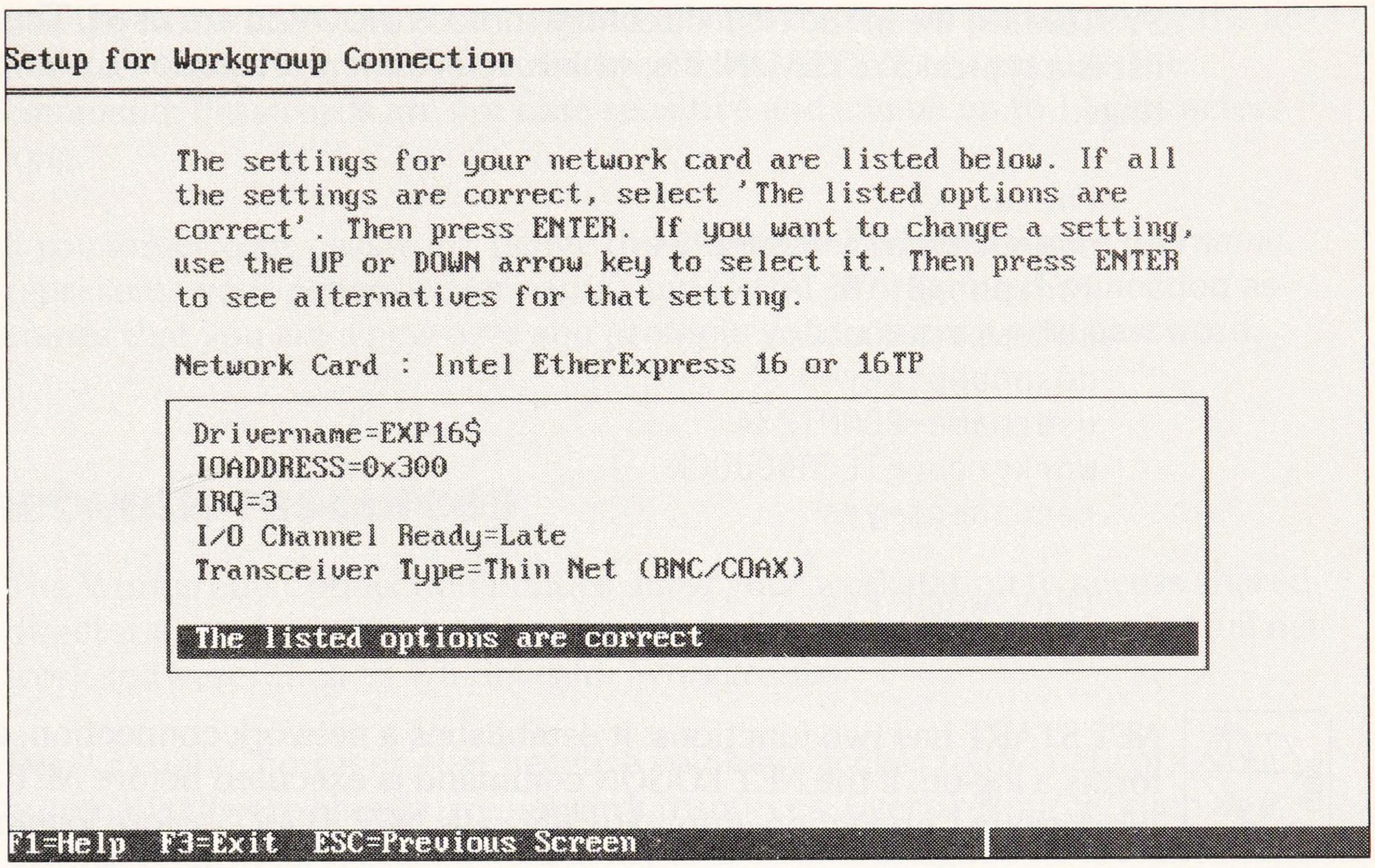
The following is a typical PROTOCOL.INI file:

```
[network.setup]
version=0x3100
netcard=ms$ee16,1,MS$EE16
transport=ms$netbeui,MS$NETBEUI
lana0=ms$ee16,1,ms$netbeui

[MS$EE16]
IOCHRDY=Late
IOADDRESS=0x300
DriverName=EXP16$
IRQ=3
TRANSCEIVER=Thin Net (BNC/COAX)

[protman]
DriverName=PROTMAN$
PRIORITY=MS$NETBEUI

[MS$NETBEUI]
DriverName=netbeui$
SESSIONS=6
NCBS=12
BINDINGS=MS$EE16
LANABASE=0
```

You can easily view PROTOCOL.INI from the File menu of MSD.

All PROTOCOL.INI files contain [NETWORK.SETUP] and [PROTMAN] sections. There are additional sections specific to the network adapter card and the network protocol — in the example, the sections are named [MS$EE16] and [MS$NETBEUI].

You should not directly edit PROTOCOL.INI. Instead, run SETUP to modify the settings. If you want to change the IRQ setting for the network adapter card, for example, run SETUP and press Enter to select the Network Card option. Then select the Edit Settings option to modify the detailed card settings (see Figure D-3).



```
Setup for Workgroup Connection

    The settings for your network card are listed below. If all
    the settings are correct, select 'The listed options are
    correct'. Then press ENTER. If you want to change a setting,
    use the UP or DOWN arrow key to select it. Then press ENTER
    to see alternatives for that setting.

    Network Card : Intel EtherExpress 16 or 16TP

     Drivername=EXP16$
     IOADDRESS=0x300
     IRQ=3
     I/O Channel Ready=Late
     Transceiver Type=Thin Net (BNC/COAX)

     The listed options are correct

F1=Help  F3=Exit  ESC=Previous Screen
```

**Figure D-3:** Use SETUP to modify network adapter card settings in PROTOCOL.INI.

## How to log on to the network

If you accept the default SETUP Startup option, which is Run Workgroup Connection and log on, the following line is added at the beginning of AUTOEXEC.BAT:

```
c:\net\net start
```

NET START is the primary command for connecting to the network. When you execute NET START without any switches, the settings located in the

### Network Protocols

The network protocol defines how data and applications messages are packaged and routed around the network. Two standard protocols are NETBIOS and NETBEUI.

NETBIOS stands for *Network Basic Input/Output System*. This system is supported by most network operating systems and by most networked applications. NETBEUI stands for *NETBIOS Extended User Interface*. Introduced in 1985 by IBM, it is a compact network protocol designed for small networks with 20 to 200 workstations.

NETBEUI is the only protocol directly supported by DOS 6. However, third-party (OEM) protocols can be installed using SETUP.

SYSTEM.INI file in the NET directory (not \WINDOWS) are used. The following list is a typical SYSTEM.INI file, which details the workstation's network settings:

```
[network]
computername=ROOM1234
lanroot=C:\NET
autostart=basic
dospophotkey=N
username=ROOM1234
workgroup=TECHNOJOCK
reconnect=yes

[Password Lists]
ROOM1234=C:\NET\ROOM1234.PWL
```

GEM

NET START has two functions: it establishes a network connection, and it performs a log-on. If the NET LOGON command is executed before NET START, Workgroup Connection automatically calls NET START before logging on.

NET START does not allow you to enter a password as a parameter, whereas NET LOGON does. If you use the NET LOGON command, specifying the password along with a /YES switch, you can automatically invoke NET START and NET LOGON without being prompted for a password. For example, change the command

```
c:\net\net start
```

to

```
c:\net\net logon room1234 abracadabra /yes
```

If you want to log on as a different user, just enter the NET LOGON command, and Workgroup Connection automatically issues a NET LOGOFF command before executing the NET LOGON command.

## Password management

The first time you log on to the network, Workgroup Connection asks you to confirm your password. The password is stored in encrypted form in the network directory, in a file with a PWL extension. The first eight characters of the filename identify the user name. A full list of PWL files and their corresponding user names is maintained in the [PASSWORD LISTS] section of SYSTEM.INI. This allows many different users to access the network from one workstation.

Use the NET PASSWORD command to change the log-on password for the active user name. You must be connected to the network to execute this command. Passwords are not case sensitive and can be up to 14 characters long.

If you forget your password, delete the *username*.PWL file (where *username* represents your network user name) and log on. Workgroup Connection assumes that you are a new user and prompts you to enter a new password.

## Using network resources

The Workgroup Connection utilities allow your workstation to access shared directories and printers attached to other computers that are running full network software, such as Windows for Workgroups.

Don't assume, however, that because you have access to the network, you automatically are able to use network resources. Every resource, such as a printer or directory, must be explicitly shared and assigned a resource name. The resource name is known as the *sharename*. In addition, some resources can only be shared when a password is entered.

### Using the Pop-Up utility

You can either use the Pop-Up program or the NET subcommand utilities to access shared resources. But the Pop-Up utility provides the easiest and most interactive way to access shared directories and printers.

#### Loading and running Pop-Up

To execute the Pop-Up program, just log on and then enter the following command:

```
net
```

You also can install the Pop-Up program as a TSR. To do so, enter the following command:

```
net start popup
```

The TSR consumes about 30K of memory. You can pop it up over Text Mode applications and at the DOS prompt by pressing Alt-N. You also can use SETUP to configure Pop-Up to use a different Alt-letter combination (see Figure D-2).

The Pop-Up utility is not designed to handle monochrome display systems. If you want to use Pop-Up on a monochrome system, use the MODE command to set the display mode to mono before you execute the NET command. The following two statements launch Pop-Up on a monochrome system:

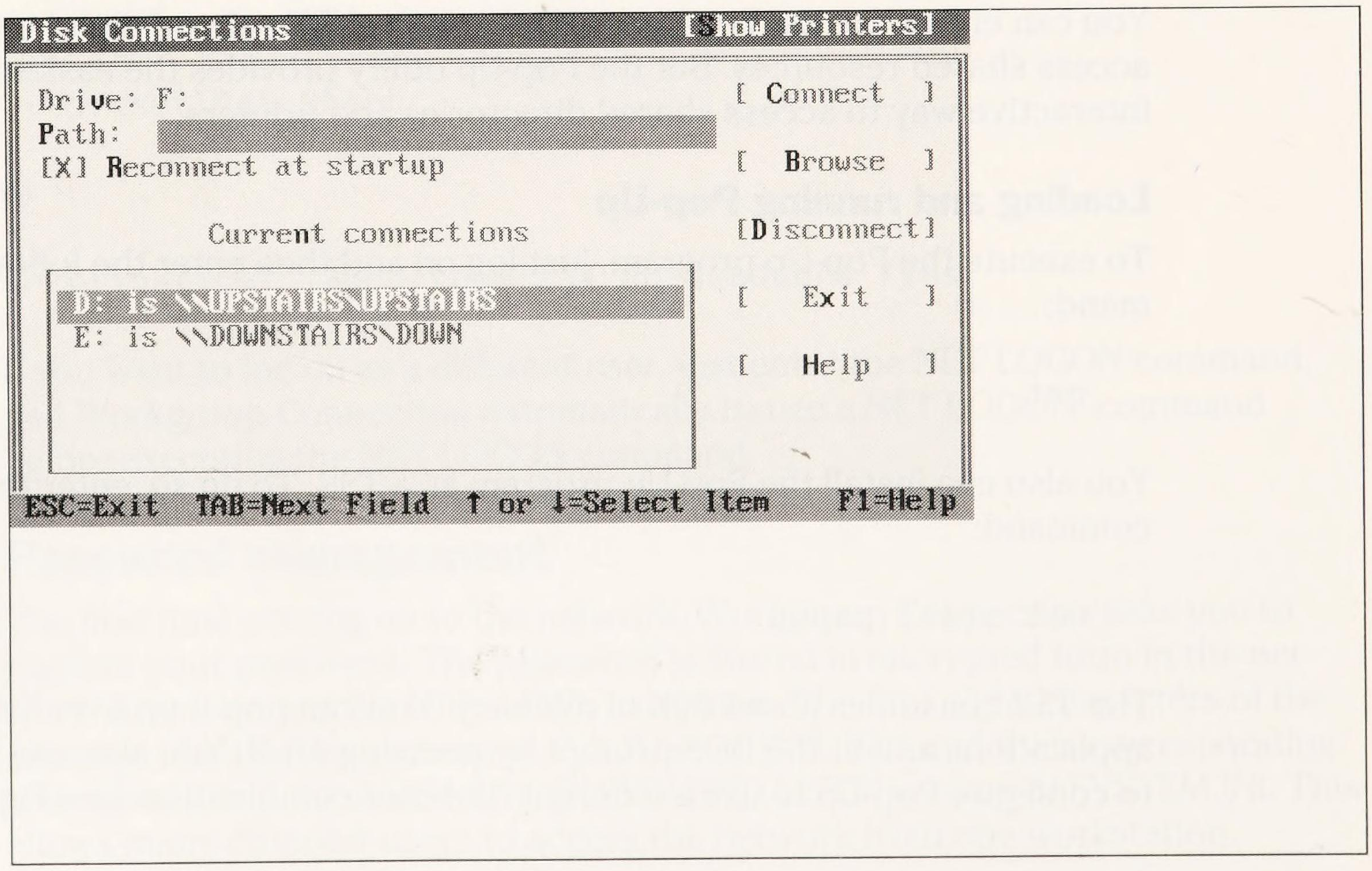
```
mode mono
net
```

You can remove the Pop-Up TSR from memory, provided that no other TSRs were loaded after it, by entering the following command:

```
net stop popup
```

Pop-Up has two main screens: one for managing remote directory connections and another for managing remote printers. Figure D-4 shows the main Disk Connections display. You can toggle between the Disk Connections and Printer Connections by pressing Alt-S (the SHOW command).

Notice that the computer names are prefixed by double backslash characters. Workgroup Connection always uses the double backslash convention to indicate a computer or workgroup name.



Drive: F:
Path:
[X] Reconnect at startup

Current connections

D: is \\UPSTAIRS\UPSTAIRS
E: is \\DOWNSTAIRS\DOWN

**Figure D-4:** Pop-Up's Disk Connections display.

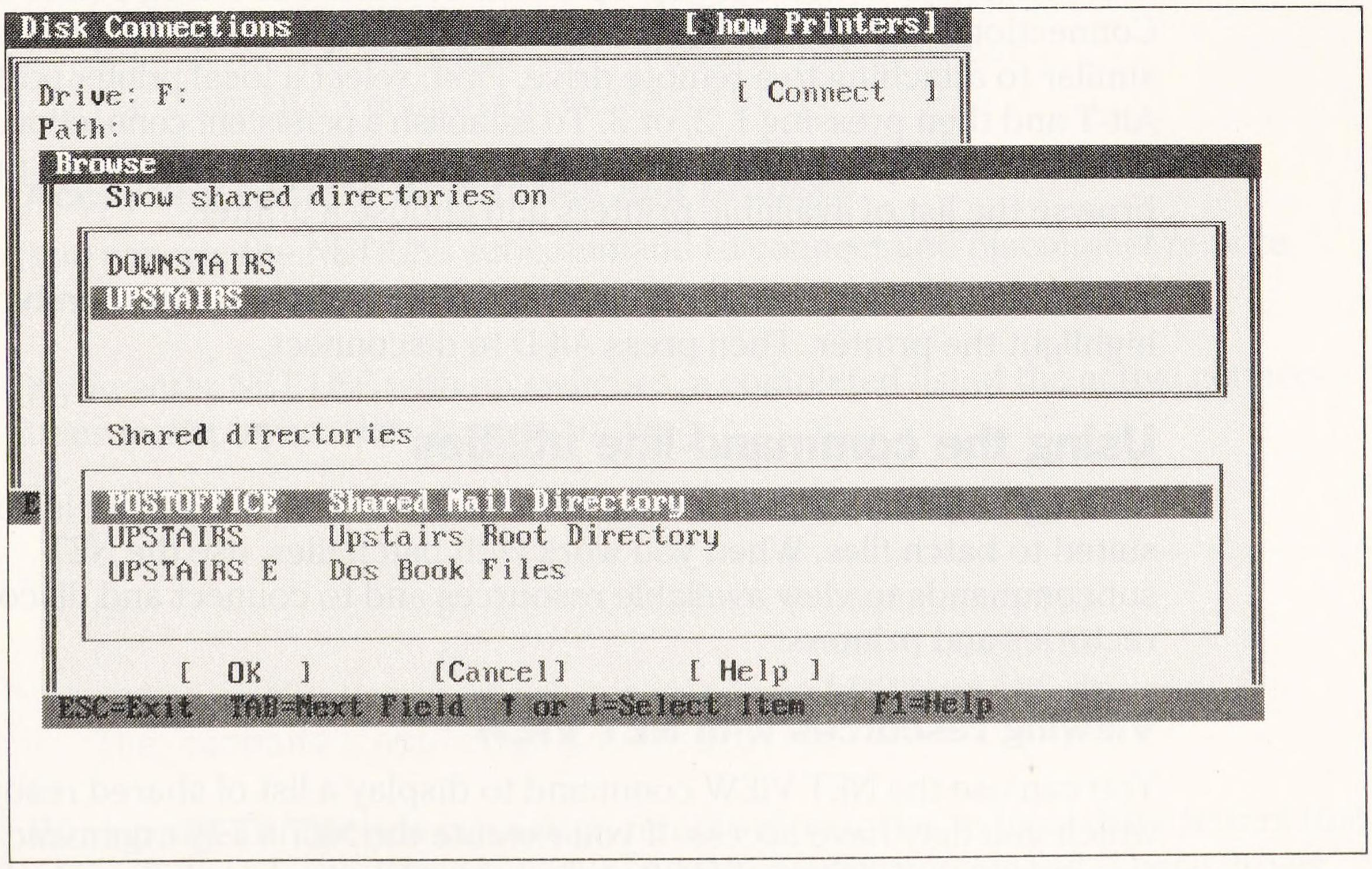
In this example, drive D is mapped to the directory with the sharename *Upstairs* on the \\UPSTAIRS computer. Drive E is mapped to the directory with the sharename *Down* on the \\DOWNSTAIRS computer. There is no way to determine precisely what directory or drive is represented by a sharename. For example, *Down* might actually be mapped to C:\THIS\THAT\THEOTHER or D:\.

### Connecting to directories

To connect to a remote drive, you can enter an explicit path in the **P**ath field, using the convention \*ComputerName**Sharename.* But the easiest method is to use the **B**rowse option. To invoke this option, press Alt-B. Pop-Up displays a list of all workstations currently connected to the network. If a particular computer name is not listed as you expected, the computer is probably turned off or not logged in to the network.

In the Browse window, use the cursor keys to highlight a workstation. A list of shared drives is displayed in the lower window. Press Tab (or Alt-D) to shift to the sharenames and select one by using the cursor keys and pressing Enter. Figure D-5 shows the list of sharenames for the shared directories on the \\UPSTAIRS computer.

By default, Pop-Up assigns the next available drive letter, but you can select any unused drive letter by pressing Alt-V before making the connection. If you want



```
Disk Connections                                 [Show Printers]
 Drive: F:                                      [ Connect  ]
 Path:
  Browse
     Show shared directories on

     DOWNSTAIRS
     UPSTAIRS

     Shared directories

     POSTOFFICE    Shared Mail Directory
     UPSTAIRS      Upstairs Root Directory
     UPSTAIRS E    Dos Book Files

          [  OK  ]         [Cancel]         [ Help ]
  ESC=Exit  TAB=Next Field  ↑ or ↓=Select Item      F1=Help
```

**Figure D-5:** Using the Browse option.

to assign a drive letter that is already in use to a new path, you must first disconnect the existing path.

If you want to connect to a remote directory every time you log on to the network, press Alt-R and then press the spacebar to toggle the check mark in the Reconnect at Startup field before you select the Browse option. Paths that are reconnected at start-up are referred to as *persistent connections*.

GEM

Persistent connections are recorded in the binary file CONNECT.DAT, which is located in the NET directory. A quick way to remove all persistent connections is to delete the file CONNECT.DAT. Alternatively, enter the following command at the DOS prompt:

```
net use /persistent:clear
```

### Connecting to shared printers

You can assign different shared printers to different local printer ports. When you try to print to the redirected printer port, the output is automatically sent to the network printer.

To use a remote printer, you must assign it to a local parallel port (LPT1 through LPT3). Use Pop-Up to establish connections to shared printers as well as to pause, resume, and delete your queued print jobs.

Press Alt-S to switch from Pop-Up's Disk Connections dialog box to the Form Connections dialog box. The process of attaching to a remote printer is very similar to attaching to a remote drive. First, select a local printer port by pressing Alt-T and then pressing 1, 2, or 3. To establish a persistent connection, press Alt-R and press the spacebar to insert a check in the Startup field. Then press Alt-B to browse the list of available printers and choose a printer.

To disconnect from a shared printer, press Alt-N and use the cursor keys to highlight the printer. Then press Alt-D to disconnect.

## Using the command-line utilities

Although the Pop-Up utility is easy to use, its interactive nature is not well suited to batch files. When you work with batch files, use the NET subcommands to view available resources and to connect and disconnect directories and printers.

### Viewing resources with NET VIEW

You can use the NET VIEW command to display a list of shared resources to which you may have access. If you execute the NET VIEW command with no parameters, a list of workstations connected to your workgroup is displayed. Here is an example:

```
Server Name                 Remark
-------------------------------------------------------------------------------
\\MCCARTP                   Room 1231
\\STARRR                    Desktop system
\\LENNONJ                   Room 1236
\\HARRISG                   What sort of remark
\\AINSBUB                   Mr Big Disk
The command completed successfully.
```

If your network has more than one workgroup, you can display a list of workstations connected to a selected workgroup. To do this, specify the workgroup name as a parameter with NET VIEW. Here is an example:

```
net view \\sales
```

To ascertain what shared resources are available at a specific workstation, enter the NET VIEW command with the name of the workstation as a parameter. If I enter the command `net view \\ainsbub`, for example, the following output is displayed:

```
Shared resources at \\AINSBUB
Sharename     Type          Comment
-------------------------------------------------------------------------------
POSTOFFICE    Disk          Shared Mail Directory
ROOTC         Disk          Drive C Root Directory
SWAPE         Disk          Directory E:\SWAP
QMS-PS        Print         Postscript printer
The command completed successfully.
```

### Accessing remote directories and printers

You can use the NET USE subcommand to connect and disconnect remote directories and printers.

If you enter NET USE with no switches, a completed list of the active connections is displayed. Here is an example:

```
Status          Local name      Remote name
-------------------------------------------------------------------------------
OK              D:              \\STARRR\LYRICS
OK              F:              \\HARRISG\SONGS
OK              LPT1            \\AINSBUB\QMS-PS
The command completed successfully.
```

NOTE

Workgroup Connection can assign remote directories to local drive letters that were allocated with Quarterdeck's QEMM Lastdrive command. These drives, however, do not show up in the NET USE list.

Refer to the Command Reference at the end of this appendix for a complete description of each of the NET USE switches and parameters. The following paragraphs provide some common examples of how you can employ the NET USE and NET PRINT commands to perform different tasks.

### Reviewing directory examples

The basic syntax for connecting with remote drives using NET USE is

```
net use drive: \\computer\path [/p:yes or no]
```

where *drive* is the drive letter that will be used to reference the directory specified in \*computer**path*. The optional /P switch can be used to control whether the connection is persistent — that is, whether the connection is to be reestablished each time you log on to the network.

To connect to a directory with sharename ROOTC on the \\AINSBUB computer as drive D, you would enter the following command:

```
net use d: \\ainsbub\rootc
```

If you want to make the drive assignment persistent (so the connection is reestablished every time you log on), you can add the /P switch. Here is an example:

```
net use d: \\ainsbub\rootc /p
```

To access a printer or directory on a computer whose name includes one or more spaces, enter the entire device name in quotes. Here is an example:

```
net use e: "\\silly name\rootc"
```

To disconnect drive D, enter the following command:

```
net use d: /delete
```

You do not have to connect to a remote directory to access its files if the directory was made shareable without a password. You can explicitly access an individual file using the following syntax:

```
\\computername\drivename\path\filename.ext
```

For example, to edit the contents of the SAYWHAT.HEY file from the computer named ROOM2554, on the drive named CDRIVE, you would enter the following command:

```
edit \\room2554\cdrive\saywhat.hey
```

### Reviewing printer examples

The basic syntax for connecting with remote printers using NET USE is

```
net use lptn \\computer\queue [/p:yes or no]
```

where `lptn` is the local printer port that will be used to direct output to the printer specified in *\\computer\queue*. The optional /P switch can be used to control whether the connection is persistent.

The following command redirects all LPT1 output to the shared printer called HPIII on the computer named \\MCCARTP:

```
net use lpt1: \\mccartp\hpiii
```

To disconnect from the remote printer assigned to LPT1, enter the following command:

```
net use lpt1: /delete
```

You can use the command NET PRINT to manage print queues after a connection with a printer is established. After you send a job to the printer, you must determine the job number before you can pause, resume, or delete the job. To see the print jobs assigned to one of your ports, enter the print command in the following format:

```
net print lptn
```

For example, to list all the print jobs queued at the printer assigned to LPT1, enter

```
net print lpt1
```

The following command pauses job 234 queued at the LPT1 printer:

```
net print lpt1 234 /pause
```

To resume printing the job, you would enter the following command:

```
net print lpt1 234 /resume
```

To delete job 667 from the LPT3 print queue, you would enter the following command:

```
net print lpt3 667 /delete
```

### Accessing CD-ROMs

You can access a remote CD-ROM drive (provided that it is shared by the owner) just as you can any other directory. The owner of the CD-ROM system must be using MSCDEX.EXE version 2.2 or later, and MSCDEX must be installed with the /S switch. Here is an example:

```
c:\dos\mscdex /s /d:mvcd001 /m:10 /v
```

You do not need to install MSCDEX on your own computer to access a remote CD-ROM drive. However, if you need MSCDEX to access a local CD-ROM drive, do not install it with an /S switch.

If a network problem occurs, Workgroup Connection displays a brief error message and an error number. To obtain a longer description of the error and suggestions about how to correct the problem, use the NET HELP command and specify the error number. For example, to obtain a full description of network error 64, you would enter the following command:

```
net help 64
```

## Mail Secrets

To use Microsoft Mail, your computer must have a network card and be connected to the network. At least one other computer in the network must be running a network operating system or a peer-to-peer network, such as Windows for Workgroups. In other words, you cannot simply connect two or more computers by using Workgroup Connection and use the Mail facility.

Using the Workgroup Connection MAIL command, you can join other Microsoft Mail users in your network and participate fully in workgroup electronic mail.

### Installing Mail

Before running Microsoft Mail for the first time, log on to the network by issuing the NET START or NET LOGON command. Then enter the following command:

```
mail
```

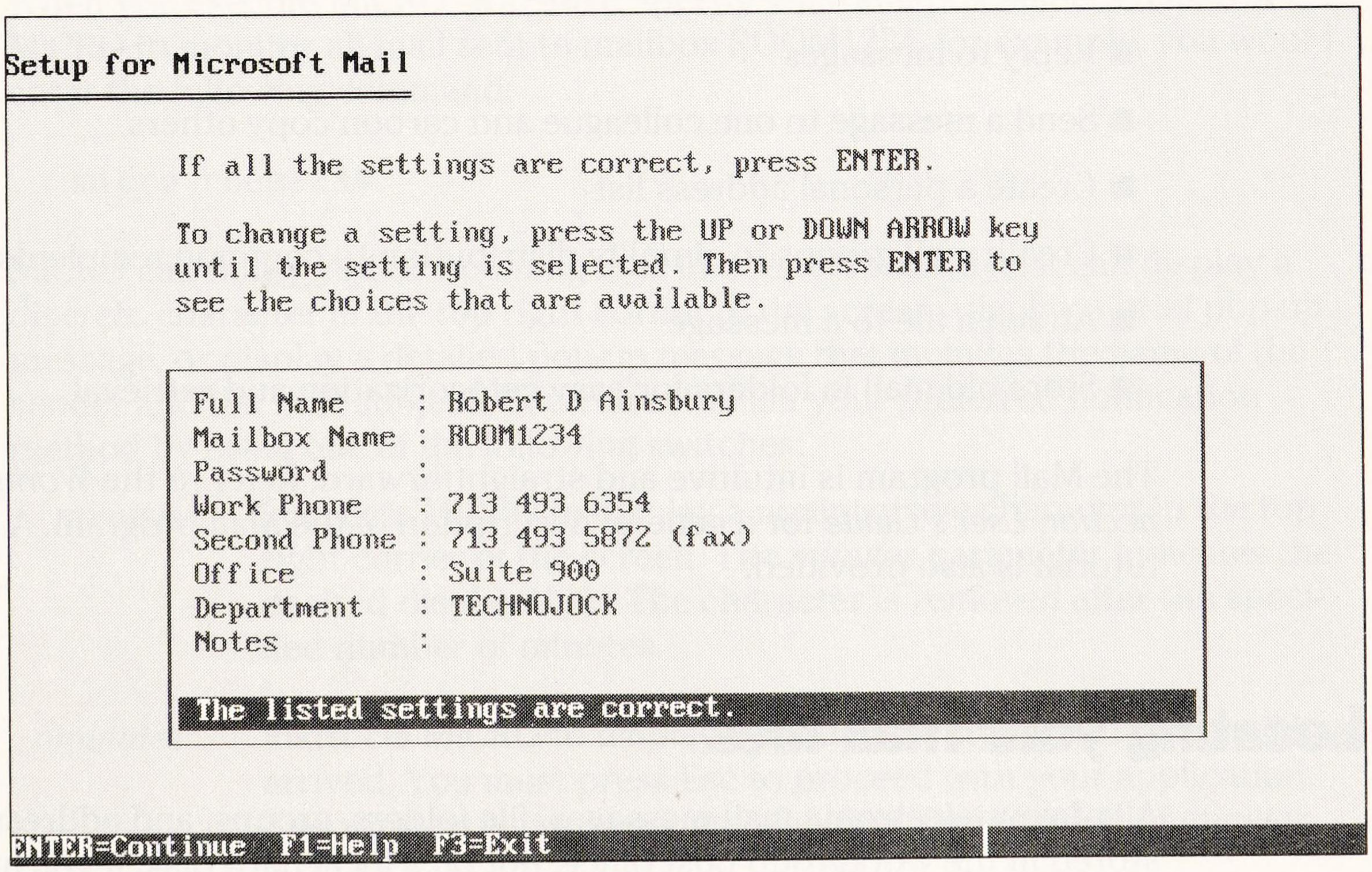
MAIL launches into Setup Mode and prompts you for the name of the workgroup post office. The workgroup post office actually is a directory located on one of the computers in your workgroup. You must enter the post office directory name in the following format:

\\*computername*\*sharename*

If you do not know the post office name, ask a network administrator or do some snooping with the NET VIEW command — by default, a Windows for Workgroups or Windows NT post office directory has the sharename (not surprisingly) *postoffice*.

During setup, MAIL prompts you to enter your name, password, telephone number, alternate telephone number, office location, and an optional note. Be aware that other network mail users are able to view these details (except for the password) when they compose mail. Figure D-6 shows the confirmation list that MAIL setup presents before creating your mailbox.

Be sure to make a note of the mailbox name. You must specify this name every time you start the Mail program. By default, your mailbox name is the same as your network user name.



```
Setup for Microsoft Mail

        If all the settings are correct, press ENTER.

        To change a setting, press the UP or DOWN ARROW key
        until the setting is selected. Then press ENTER to
        see the choices that are available.

         Full Name     : Robert D Ainsbury
         Mailbox Name  : ROOM1234
         Password      :
         Work Phone    : 713 493 6354
         Second Phone  : 713 493 5872 (fax)
         Office        : Suite 900
         Department    : TECHNOJOCK
         Notes         :

         The listed settings are correct.

ENTER=Continue  F1=Help  F3=Exit
```

**Figure D-6:** The MAIL setup confirmation list.

## Running Microsoft Mail

To run Microsoft Mail, just log on to the network and enter the MAIL command. You are then prompted for your mailbox name and your password. Alternatively, you can specify the mailbox and password (using the -P switch) as command-line parameters. For example, to use mailbox CLINTOW with the password PEROT, you can go directly to the Mail main menu by using the following command:

```
mail clintow -pperot
```

During setup, MAIL creates a persistent connection. By default, it assigns drive M to the workgroup post office directory. If you connect to the post office directory using another drive, you must use the -D switch when you execute MAIL. The switch informs Workgroup Connection of the post office drive. For example, if the post office is assigned to drive F, you use the following command to start Mail:

```
mail clintow -pperot -df
```

Microsoft Mail is a full-featured electronic mail program that enables you to perform the following tasks:

- Send, receive, and print messages
- Reply to messages
- Send a message to one colleague and carbon copy others
- Create a personal address list
- Create mail groups to simplify distribution to common recipients
- Attach a file to a message
- Store old mail in folders for easy categorization and retrieval

The Mail program is intuitive and straightforward. Refer to the *Workgroup Connection User's Guide* for a general description of the Mail program. A useful tutorial is also provided.

## Locating your mail files

All of your electronic mail messages, file folders, groups, and address lists are stored in the workgroup post office, not on a local hard disk. If you are not logged on to the network and connected to the main post office directory, you are not able to use MAIL. You are not even able to read old messages or prepare new messages for distribution.

All mail files are stored in encrypted form. Their contents cannot be readily deciphered.

## Receiving new mail

By default, MAIL constantly checks for new incoming messages whenever the program is running. You can use the -X switch to instruct MAIL to use a polling method instead. When you use this switch, MAIL defaults to checking for new messages every 30 seconds, but you can specify any polling period from 0 to 199 seconds by including the -N switch. You can only use the -N switch when -X is also specified.

The following command starts MAIL, instructs it to use the polling method, and specifies a polling period of 60 seconds:

```
mail -x -n60
```

Remember, MAIL only checks for messages when it is running. It is not a TSR. You should install the Workgroup Connection program MICRO if you want to be informed of incoming mail while you are running another application, such as a word processor or spreadsheet program.

When you execute MICRO, you must specify your mailbox name. To instruct MICRO to monitor all mail sent to mailbox ROOM1234, for example, you would enter the following command:

```
micro room1234
```

MICRO can notify you of incoming mail in one of three ways. It can display a discrete character in the top right corner of the screen, display a brief pop-up message, or display a detailed pop-up message that includes the name of the sender and the mail subject. You can establish your preferred notification method by using one of the following switches:

| | |
|---|---|
| `-f`*minutes* | Instructs MICRO to display a nonintrusive character in the top right corner of the screen. The *minutes* parameter indicates the desired display time. The character is removed after the specified number of minutes. |
| `-n`*minutes* | Instructs MICRO to display a window notifying you that mail has arrived. You must press Esc to proceed with your application. The window is removed after the specified number of *minutes*. |

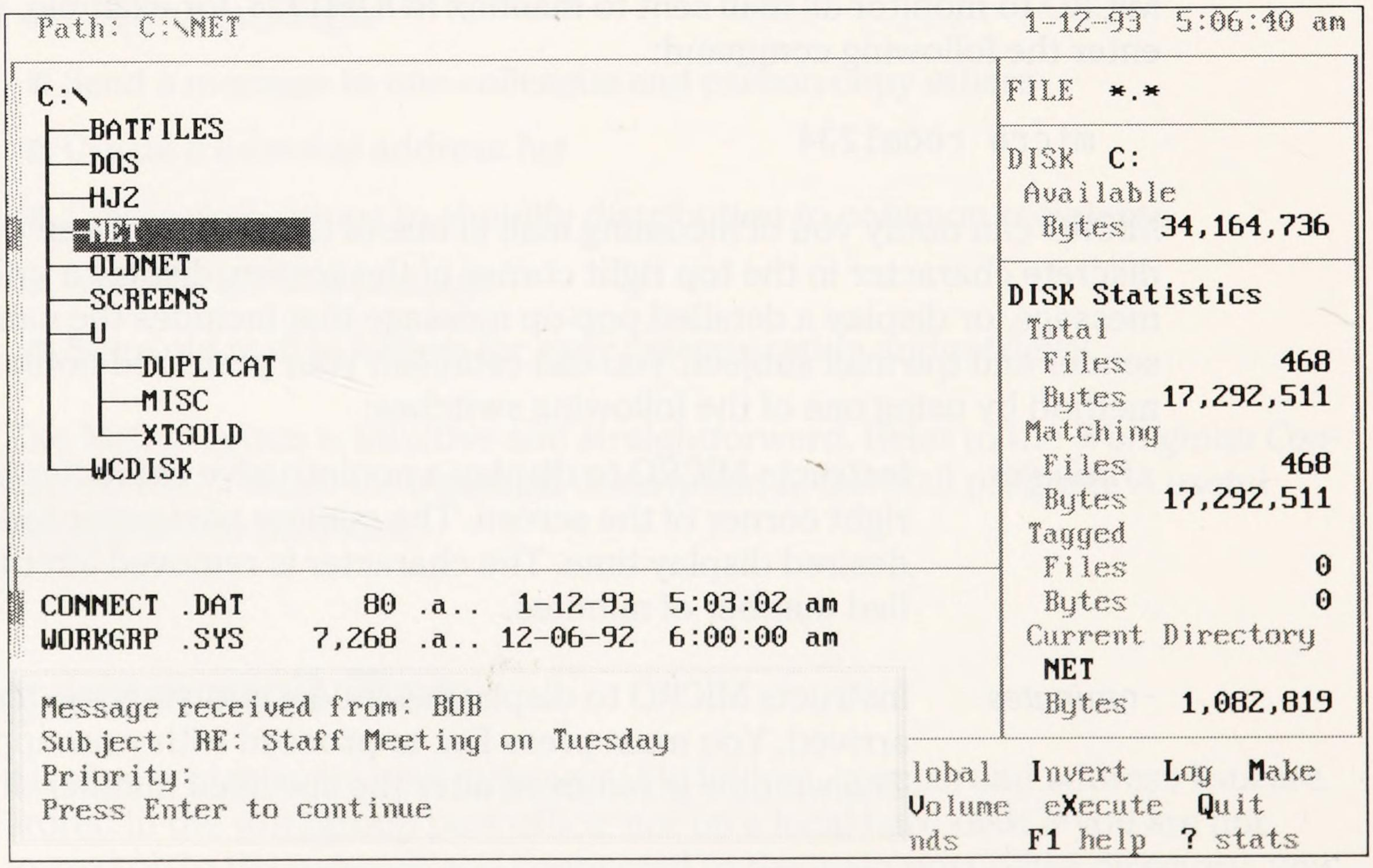
If you will be running graphics applications, such as Windows or the graphics version of DOS Shell, use the -F switch. In other words, instruct MICRO to display the special notification character rather than a pop-up window. MICRO is unable to successfully display pop-up messages when your system is in Graphics Mode.

If neither switch is specified when MICRO is installed, a setting of -N1 is used. If the -N switch is in effect, you can also use the -P switch, instructing MICRO to provide information about the message (the name of the sender and the subject of the message). If your mailbox is password protected, you must specify your password. The following command installs MICRO, instructing it to poll the mailbox ROOM1234 once every two minutes and display a detailed message when mail is received:

```
micro room1234 -n2 -pabrcadabra
```

In Figure D-7, you see a message from MICRO in the lower left corner of the screen. MICRO indicates that BOB sent a mail message about Tuesday's staff meeting.

If you use the -F switch, which tells MICRO to use a nonintrusive character to alert you to incoming mail, you can remove the indicator by pressing Alt-F1.



```
Path: C:\NET                                      1-12-93   5:06:40 am

C:\                                           FILE    *.*
├──BATFILES
├──DOS                                        DISK   C:
├──HJ2                                          Available
├──NET                                           Bytes   34,164,736
├──OLDNET
├──SCREENS                                    DISK Statistics
├──U                                            Total
│   ├──DUPLICAT                                  Files          468
│   ├──MISC                                      Bytes   17,292,511
│   └──XTGOLD                                  Matching
└──WCDISK                                        Files          468
                                                 Bytes   17,292,511
                                                Tagged
                                                 Files            0
                                                 Bytes            0
CONNECT  .DAT       80 .a..  1-12-93  5:03:02 am  Current Directory
WORKGRP  .SYS    7,268 .a.. 12-06-92  6:00:00 am   NET
                                                 Bytes    1,082,819
Message received from: BOB
Subject: RE: Staff Meeting on Tuesday          lobal   Invert   Log   Make
Priority:                                      Volume  eXecute  Quit
Press Enter to continue                        nds     F1 help  ? stats
```

**Figure D-7:** A sample MICRO message.

MICRO consumes about 13K of RAM. You can unload MICRO from memory (provided that no other TSRs were loaded after it) by specifying the -U switch as follows:

```
micro -u
```

Refer to Part IV, "The Complete DOS 6 Command Reference," for a complete description of all MICRO switches.

If you execute MAIL while MICRO is installed, specify the -M switch. MAIL leaves it up to MICRO to inform you about incoming messages. Here's an example:

```
mail clintow -pperot -df -m
```

## Forgetting your password or mailbox name

Every workgroup post office has a designated mail administrator — someone responsible for managing the system. If you forget your password, the administrator can come to your rescue. The administrator cannot tell you the old password but can overwrite your old password with a new one.

The administrator can also tell you your mailbox name and optionally modify your mailbox settings (such as phone numbers, office location, and so on).

## Workgroup Connection Command Reference

This section is a summary of the following Workgroup Connection programs:

- MAIL
- MICRO
- NET
- NET CONFIG
- NET LOGOFF
- NET LOGON
- NET PASSWORD
- NET PRINT
- NET START
- NET STOP
- NET TIME
- NET USE
- NET VER
- NET VIEW

**CROSS REFERENCE**

Refer to the beginning of Part IV for a thorough description of the formatand syntax conventions used in this Command Reference.

BOB'S SCORECARD 7

# MAIL

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ☒ | ■ | ☒ | ☐ |

## Description

This command installs Microsoft Mail, the DOS 6 electronic-mail program.

## Usage

mail [*mailboxname*] [-*lines*] [-c] [-d*n*] [-h] [-m] [-n*seconds*] [-p*password*] [-s*display*] [-v] [-w*color*] [-x]

## Options and switches

***Note:*** The first time you try to run the MAIL program, DOS invokes the MAIL setup program.

When you execute MAIL with no switches, you are prompted for a mailbox name (assumed to be on drive M), and — if all is well — the main menu appears.

[*mailboxname*]

This option specifies your personal mailbox name. Every mail user in the workgroup has a mailbox (in a dedicated directory). If the mailbox is not specified, MAIL prompts you for it. If you do not use the –D switch, MAIL assumes that the mailbox is located on drive M.

[-*lines*]

This option specifies the number of display lines to use. EGA systems support 43 lines, and VGA systems support 43 and 50 lines. By default, 25 lines are used.

[-c]

On some (obscure) color systems, because MAIL thinks that it is running on a monochrome system, it uses only black and white. Use the –C switch to instruct MAIL to use colors.

`[-dn]`

*n* specifies the drive that contains the workgroup mailbox. Any letter (from A to Z) is accepted, and M is the default.

`[-h]`

This switch displays the command-line help; it has the same effect as /?.

`[-m]`

This switch informs MAIL that, because MICRO (the mail notifier) is loaded, MAIL does not need to signal when new mail is received.

`[-nseconds]`

Used with the –X switch, –N specifies the polling frequency, in seconds, to check for new mail. MAIL checks for new messages every 30 seconds by default, but any value in the range from 0 to 199 is accepted. A value of 0 disables polling altogether.

`[-ppassword]`

This switch identifies the mailbox password. If the switch is not specified, MAIL prompts for the password. If your mail is confidential in any way, do not record the password in a batch file.

`[-sdisplay]`

Use this switch when MAIL cannot detect the video system. Valid values for display are MONO, HERC, CGA, EGA, and VGA.

`[-v]`

This switch suppresses screen flicker on CGA systems.

`[-wcolor]`

This switch specifies the border color. Any value in the range from 0 (black) to 15 (white) is supported.

`[-x]`

This switch instructs MAIL to poll for new messages rather than continually check for new mail.

### Remarks

To use Microsoft Mail, your computer must have a network card and be connected to the network. At least one other computer in the network must be running a network operating system or a peer-to-peer network, such as Windows for Workgroups. In other words, you cannot simply connect two or more computers using DOS 6 and use the MAIL program.

Each user has an individual mailbox in a *postoffice* located on one of the computers running the network software. If you are unsure of the mailbox name, contact your network administrator.

### Examples

To start MAIL by using the mailbox BOBBY when the postoffice is located on drive F, enter the following command:

```
mail bobby /df
```

To start MAIL and poll for messages every 60 seconds, enter the following command:

```
mail -x -n60
```

### Additional information

CROSS REFERENCE

*See also* MICRO and NET.

## MICRO

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 6 | ■ | ■ | ☒ | ☐ |

### Description

This command is a small TSR that checks your mailbox and notifies you when new mail is received.

### Usage

`micro` *mailboxname* `[-c]` `[-d`*n*`]` `[-f`*minutes*`]` `[-n`*minutes*`]` `[-p`*password*`]` `[-u]` `[-x`*seconds*`]`

### Options and switches

*mailboxname*

This option is the name of your mailbox.

```
[-c]
```

This switch forces MICRO to use colored text, even when it thinks it is running on a monochrome system.

[-d*n*]

*n* specifies the drive that contains the workgroup mailbox. Any letter (from A to Z) is accepted, and M is the default.

[-f*minutes*]

This switch instructs MICRO to display a (nonintrusive) character in the top right corner of the display. It makes you aware that mail has arrived but enables you to proceed with your application. The character is removed after the specified number of *minutes*.

[-n*minutes*]

This switch instructs MICRO to pop up a window to notify you that mail has arrived. You must press Esc to proceed with your application. The window is removed after the specified number of *minutes*. If neither the –F or –N parameter is specified, a setting of –N1 is applied.

[-p*password*]

This switch instructs MICRO to provide information about the mail (who sent the mail and the subject). If your mailbox is password protected, you must specify your password.

```
[-u]
```

This switch unloads MICRO from memory, assuming that no other TSRs were loaded after MICRO.

[-x*seconds*]

This switch instructs MICRO to poll for new messages one time every specified number of *seconds* rather than constantly check for new mail. The value of *seconds* can be any number in the range from 1 to 3600.

## Remarks

You must create a mailbox with MAIL before you can run MICRO.

You can remove the mail notifier before the expiration time by pressing Alt-F1.

If you are running graphics applications, such as Windows or the graphics version of DOSSHELL, use the /F switch.

### Examples

To install MICRO to monitor Buster's mail and instruct MICRO to display the discrete mail notifier for five minutes, use the following command:

```
micro buster -f5
```

To install MICRO to monitor BClinton's mail when the mailbox is situated on the J drive, enter the following command:

```
micro bclinton -dj
```

### Additional information

CROSS REFERENCE

*See also* MAIL.

## NET

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 8 | ☒ | ☒ | ☒ | ☐ |

### Description

Provides DOS with workgroup (network) capabilities. The NET family of commands can be used only on systems that have a network card installed and that are connected to a Windows for Workgroups or Microsoft LAN Manager (or compatible) network.

When you execute NET with no switches, the network pop-up utility is displayed for configuring drives and printers.

For clarity, the NET command has been divided into the following subcommands:

NET CONFIG
NET LOGOFF
NET LOGON
NET PASSWORD
NET PRINT
NET START
NET STOP
NET TIME

NET USE
NET VER
NET VIEW

To take advantage of any of the NET commands, you should have the network device drivers installed in CONFIG.SYS, as shown in the following examples:

```
device=c:\net\protocol.sys /i:c:\net
device=c:\net\workgrp.sys
device=c:\net\exp16.dos
```

*Note:* The last device driver is a network card driver. The Workgroup Connection program SETUP installs the appropriate driver after determining the type and brand of network card.

On most systems, you then establish a network connection by entering the command NET START or adding it to AUTOEXEC.BAT. If the following message appears:

```
Error 3653: The protocol manager could not be found
```

check CONFIG.SYS to make sure that the network drivers are installed.

After you enter your username and password, any persistent connections are established. Bingo, you're connected.

The HELP switch for the NET command has the same effect as /?. Optionally, you can specify an error message or subcommand (NET HELP LOGOFF or NET HELP 3653, for example).

| | BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|---|
| **NET CONFIG** | 6 | ■ | ☒ | ☒ | ☐ |

## Description

This command shows the active workgroup settings.

## Usage

```
net config [/yes]
```

## Options and switches

```
[/yes]
```

This optional switch suppresses any prompts for confirmation. Normally, however, there are no prompts.

### Remarks

The settings displayed include the computer name, username, software version, workstation directory, and workgroup.

The computer name always has a prefix of two backslashes (\\BUSTER, for example). This convention is used when you want to access a remote drive or printer.

### Example

To display the workgroup settings, enter the following command:

```
net config
```

## NET LOGOFF

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 6 | ■ | ☒ | ☒ | ☐ |

### Description

This command disconnects your computer from the workgroup.

### Usage

```
net logoff [/yes]
```

### Options and switches

```
[/yes]
```

This switch bypasses the confirmation prompt.

### Remarks

During the log-off operation, NET displays all the connections to disk drives and printers that are being disconnected.

Logging off does not unload network device drivers from memory.

Use this command to disconnect from the workgroup and log on as a different user.

### Example

Enter the following command to disconnect your workstation from the workgroup:

```
net logoff
```

## NET LOGON

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

### Description

This command connects your workstation to a workgroup.

### Usage

```
net logon [username [password or ?]] [/yes]
```

### Options and switches

If you execute NET LOGON with no other parameters, you are prompted to enter your username and password.

```
[username]
```

This option specifies the username you want to use to log on.

```
[password or ?]
```

This option specifies the password for the username ID. If you use ? as the password, NET prompts for the password even if you use the /YES switch.

```
[/yes]
```

This switch instructs NET to bypass any confirmation prompts.

### Remarks

If you never have logged on as *username,* NET asks you to confirm your password.

The NET START command automatically invokes NET LOGON after the redirector has been loaded. If you try to run NET LOGON before NET START, NET automatically invokes NET START first.

If you are already logged on when you enter the NET LOGON command, NET executes a LOGOFF command.

When you first connect to a drive or printer with NET USE you can optionally specify that the connection be *persistent.* This option saves you from having to connect to remote drives and printers every time you log on. When you log on, any persistent connections are automatically reestablished.

## Examples

To log on to the workgroup as SUGAR, enter the following command:

```
net logon sugar
```

To log on as SPIKE by using the password SESAME, enter the following command:

```
net logon spike sesame
```

Each username password is stored in a special encrypted file. The file is named with the convention *username*.PWL and is located in the NET directory.

If you forget your password, delete the *username*.PWL file and log on. NET thinks that you are a new user and prompts for a new password. Voilà, you are back in business.

The NET START command does not enable you to enter a password as a parameter. If you use the NET LOGON command and specify the password and a /YES switch, you can automatically invoke NET START and log on without being prompted for a password. For example, change the following command:

```
net start
```

to

```
net logon buster abracadabra /yes
```

## Additional information

*See also* NET START and NET LOGOFF.

## NET PASSWORD

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 7 | ■ | ☒ | ■ | ☐ |

### Description

This command changes the log-on password for the active username.

### Usage

```
net password [oldpassword [newpassword]]
```

### Options and switches

[*oldpassword*]

This option specifies the existing password.

[*newpassword*]

This option specifies the new password, which can be as long as 14 characters.

### Remarks

The password is stored in a special encrypted file in the NET directory. The filename begins with the first eight characters of the username and has a PWL extension. If you forget your password, simply delete this file.

The password is not case sensitive.

### Examples

To be prompted through the password-changing process, enter the following command:

```
net password
```

To change the password from SUPERCALI to FRAGILISTIC, enter the following command:

```
net password supercali fragilistic
```

### Extended use

You can use the NET PASSWORD command also to change the password on a Microsoft LAN Manager server or domain. Enter the command HELP NET PASSWORD for more information.

## NET PRINT

BOB'S SCORECARD 7

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

This command manages shared printers in the workgroup and the workstation's print jobs.

### Usage

```
net print \\computername[\queue] or lptn [/yes]
```

or

```
net print \\computername or lptn [job# [/pause or /resume or
/delete]] [/yes]
```

### Options and switches

`\\computername`

This option specifies the workgroup computer name whose print queues are polled.

`[\queue]`

This option specifies the name of the printer queue.

`lptn`

This option specifies the name of the local parallel port whose data is redirected to the network printer.

`[job#]`

This option specifies the number of a specific print job.

```
[/pause]
```

This switch pauses a job that is being printed.

```
[/resume]
```

This switch resumes the printing of a job that was suspended.

```
[/delete]
```

This switch stops a job from being printed.

```
[/yes]
```

This switch bypasses any NET PRINT confirmation prompts.

## Remarks

All workgroup computers and shared printers are assigned an alias or name. Use the NET USE command to establish a connection to a shared workgroup printer.

You can assign different shared printers to different local printer ports. When you try to print anything to the redirected printer port, the output automatically is sent to the network printer.

After you have sent a job to the printer, you must determine the job number before you can pause, resume, or delete the job. To see the print jobs assigned to one of your ports, enter the print command in the following format:

```
net print lptn
```

You cannot share your printer or printers with other computers in the workgroup, unless, of course, you install a network program, such as Windows for Workgroups.

Other users in the workgroup who are running Windows for Workgroups can make their printers available by using the Printer Share option in Print Manager.

## Examples

If you are connected to two laser printers on the workstation named ROOM2332, you can see which printer has the least number of outstanding jobs by entering the following command:

```
net print \\room2332
```

To see how many jobs you have waiting when you have connected the LPT1 port to a shared printer, enter the following command:

```
net print lpt1
```

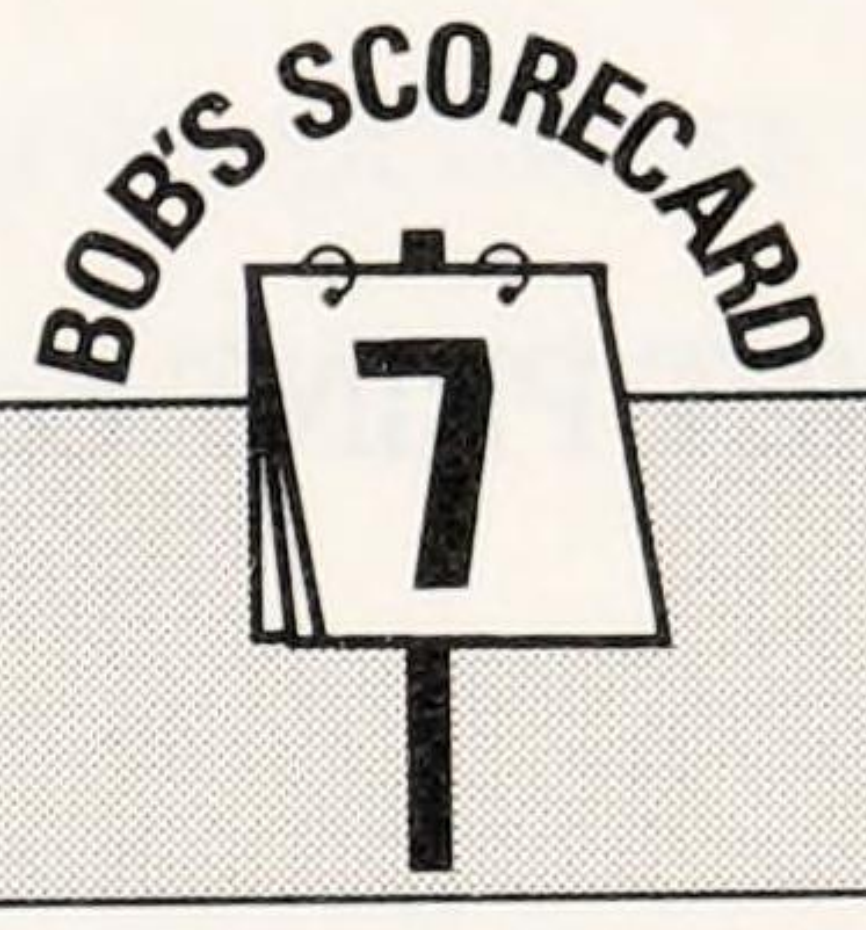
To cancel job number 445, enter the following command:

```
net print \\room2332 445 /delete
```

### Additional information

The NET USE command establishes a connection with a shared printer.

## NET START



| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ■ | ■ | ☐ |

### Description

This command establishes a workgroup connection and optionally loads the pop-up utility to simplify the creation of drive and printer connections.

### Usage

```
net start [popup or basic or full or workstation or netbind
or netbeui] [/list] [/yes]
```

### Options and switches

When you execute NET START with no switches, the settings located in the SYSTEM.INI file in the NET directory are used. This file was created by WCSETUP.

```
[popup]
```

This option loads the pop-up utility as a TSR.

```
[basic]
```

This option starts the basic redirector.

```
[full]
```

This option starts the full redirector.

```
[workstation]
```

This option starts whichever redirector was specified when WCSETUP was first run.

```
[netbind]
```

This option binds network protocols and drivers.

```
[netbeui]
```

This option starts the generic NetBEUI interface.

```
[/list]
```

This switch displays the active network settings.

```
[/yes]
```

This switch bypasses the standard confirmation prompts.

## Remarks

Don't run NET START from within a multitasking session, such as DOSSHELL or Windows.

NetBIOS, or *Net*work *B*asic *I*nput/*O*utput *S*ystem, is supported by most network operating systems and most networked applications.

NetBEUI, or *NetB*IOS *E*xtended *U*ser *I*nterface, is a compact network protocol designed for small networks with between 20 and 200 workstations.

NetBIOS and NetBEUI are supported by Windows for Workgroups.

NetBEUI generally is faster and more efficient than NetBIOS, but it is not as widely adopted. WCSETUP configures NET START to default to the appropriate interface.

## Examples

To use the defaults from SYSTEM.INI to start the workgroup connection, enter the following command:

```
net start
```

To install the pop-up TSR, enter the following command:

```
net start popup
```

CROSS REFERENCE

### Additional information

*See also* NET LOGON and NET STOP.

## NET STOP

BOB'S SCORECARD 6

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☐ | ☒ | ☐ |

### Description

This command disconnects a workstation and optionally unloads the pop-up TSR.

### Usage

```
net stop [popup or basic or full or workstation or netbeui]
[/yes]
```

### Options and switches

`[popup]`

This option unloads the pop-up utility but leaves the workgroup connection intact.

`[basic]`

This option stops the basic redirector.

`[full]`

This option stops the full redirector.

`[workstation]`

This option stops whichever redirector is named in SYSTEM.INI.

`[netbeui]`

This option stops the NetBEUI interface.

`[/yes]`

This option bypasses the standard confirmation prompts.

### Remarks

The most common use of NET STOP is to unload the pop-up utility.

Although you can use NET STOP before you reexecute NET START with new parameters, you rarely will want to do so.

### Examples

To unload the pop-up utility, enter the following command:

```
net stop popup
```

Enter the following command to log off and disconnect from the workgroup:

```
net stop
```

## NET TIME

| BOB'S SCORECARD | PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|---|
| 5 | ■ | ■ | ■ | ☐ |

### Description

This command synchronizes your computer's date and time with the network's.

### Usage

```
net time [\\computer or /workgroup:wgname] [/set] [/yes]
```

### Options and switches

```
[\\computer]
```

This option specifies the name of the time server computer with which you want to synchronize your date and time.

```
[/workgroup:wgname]
```

This option specifies that you want to synchronize with a time server computer on another workgroup.

```
[/set]
```

This switch automatically synchronizes your computer.

```
[/yes]
```

This switch bypasses the standard confirmation prompts.

### Remarks

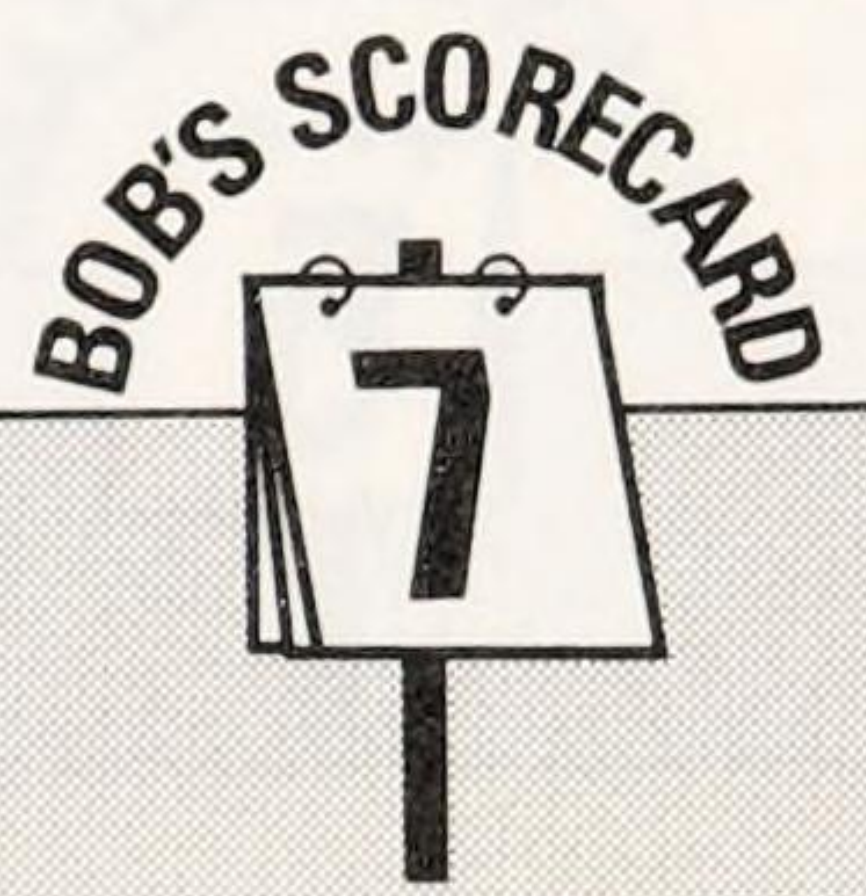
This command works only on networks that use Microsoft LAN Manager and that have a designated time server. It does not function with Windows for Workgroups.

### Example

The following command synchronizes the local system time with the time on the workstation BIGBEN:

```
net time \\bigben /set
```

## NET USE



| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

This command connects or disconnects network drives and printers.

### Usage

NET USE supports the following uses:

```
net use [drive: or *] [\\computer\path [password or ?]]
[/persistent:yes or no] [/savepw:no] [/yes]
```

```
net use [lptn] [\\computer\queue [password or ?]]
[/persistent:yes or no] [/savepw:no] [/yes]
```

```
net use drive: or \\computer\path or * /delete [/yes]
```

```
net use lptn or \\computer\queue /delete [/yes]
```

```
net use /persistent:yes or no or list or save or clear
[/yes]
```

### Options and switches

```
[drive:]
```

This option specifies the drive letter of the shared drive.

`*`

When you use * to create a link with a remote directory, * indicates that the next available drive letter should be used. When * is used with the /DELETE switch, however, all printer and drive connections are removed.

`lpt`*n*

This option specifies the name of the local printer port (LPT1, LPT2, or LPT3).

`\\`*computer*

This option specifies the name of the computer sharing the printer or drive.

*path*

This option specifies the path on the remote computer accessed by way of a local drive letter.

*queue*

This option specifies the name of the printer attached to the remote computer.

*password* or ?

If the owner of the shared resource (printer or drive) has password-protected it, you can specify the password or be prompted for it by adding using the ? character.

`/delete`

This switch disconnects the specified drive or drives or the specified printer or printers.

`/persistent:yes`

This switch instructs NET to store the connections in the network INI file so that the connections are established automatically every time you log on from this workstation. All subsequent connections in the same session also are persistent until another persistent switch is used.

`/persistent:no`

This switch specifies that the current connection and all subsequent connections are effective for only the current session.

```
/persistent:save
```

This switch makes all active connections persistent.

```
/persistent:list
```

This switch displays a list of all persistent connections.

```
/persistent:clear
```

This switch clears all persistent connections and deactivates them.

```
/savepw:no
```

This switch instructs NET USE not to record the password. This switch ensures that persistent connections to password-protected resources always prompt for the password.

```
[/yes]
```

This switch bypasses the standard confirmation prompts.

## Remarks

NET USE is much easier to use than it is to define.

By using the /PERSISTENT switch, you can automate the drive- and printer-connection procedure so that the same connections are established every time you log on. All connections are persistent by default. After you have established your standard connections and made them persistent, you can turn off the persistent setting by using the command NET USE /PERSISTENT:NO.

Thankfully, the /PERSISTENT switch can be abbreviated as /P.

If an annoying colleague has assigned a computer a name that includes a space, the drive or print queue name must be included in quotes (`"\\silly name\root"`, for example).

Use the NET VIEW command to determine computer names, shared drive names, and shared printer names.

## Examples

To connect the local drive G with the shared directory COMMON on the computer named BUSTER, enter the following command:

```
net use g: \\buster\common
```

To redirect all LPT1 output to the shared printer and call POSTSCRIPT on the computer named BOSS, enter the following command:

```
net use lpt1: \\boss\postscript
```

To disconnect drive Z from a remote computer, enter the following command:

```
net use Z: /delete
```

To see a list of all active connections, enter the following command:

```
net use
```

To make all the active connections persistent, enter the following command:

```
net use /p:save
```

To see a list of all persistent connections, enter this command:

```
net use /p:list
```

To erase all persistent connections permanently, enter the following command:

```
net use /p:clear
```

## Additional information

You can use the Pop-Up network utility to manage shared resources.

You do not have to connect to a drive to access files on another workgroup computer. If the specified path has been made shareable without a password, you can explicitly access an individual file by using the following syntax:

*\\computername\drivename\path\filename.ext*

To list the contents, for example, of the CONFIG.SYS file from the computer named ROOM2554 on the drive named CDRIVE, enter the following command:

```
type \\room2554\cdrive\config.sys
```

## NET VER

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

This command displays the version number of the network redirector.

### Usage

```
net ver
```

### Options and switches

None.

### Remarks

Enough said.

## NET VIEW

BOB'S SCORECARD 5

| PROMPT | AUTOEXEC | BATCH | CONFIG |
|---|---|---|---|
| ■ | ☒ | ☒ | ☐ |

### Description

This command provides an easy way to determine the computer names of other computers in the workgroup and the directories and printers they made shareable.

### Usage

```
net view [\\computername] [/yes]
```

or

```
net view [/workgroup:wgname] [/yes]
```

### Options and switches

When you execute NET VIEW with no switches, a list of all the computer names actively connected in the workgroup is displayed.

```
[\\computername]
```

When the name of a specific computer is specified, a list of all shared resources is displayed.

```
[/workgroup:wgname]
```

This switch instructs NET VIEW to display the computer names for a different workgroup.

```
[/yes]
```

This switch bypasses the standard confirmation prompts.

## Remarks

Only computers connected to the network at the time you issue the NET VIEW command are listed.

## Examples

To see a list of all computers connected to your workgroup, enter the following command:

```
net view
```

To see a list of all the resources that can be shared on the computer named BUSTER, enter the following command:

```
net view \\buster
```

# Appendix E

# Additional DOS Information Sources

Brilliant though this book may be (!), you can use many other valuable sources of information to keep you primed as a bona fide DOS Jedi. This appendix includes some excellent information sources. It is organized into the following categories:

- Microsoft telephone support
- Magazines and periodicals
- PC user groups
- On-line sources
- Information for users with disabilities

## Microsoft Telephone Support

If you can't find the answers to your questions in this book, in the *MS-DOS User's Guide*, or in the TXT files located in the DOS directory, you can call Microsoft at 1-206-646-5104. Be aware that 90 days after your first call, however, you will only be able to get voice support on DOS 6 directly from Microsoft by using the 900 service at 1-900-555-2000, at a cost of $2 per minute. Rather than call the 900 service, you can pay a flat rate of $25 per call (!) by calling 1-206-646-5108.

## Magazines and Periodicals

*The DOS Resource Guide,* an IDG publication, is a quarterly gem that is packed with contemporary information about DOS and general PC utilities. I first discovered it in my grocery store and have been a regular reader ever since.

For all the latest scoop on the PC scene, I recommend that you subscribe to *PC World,* published by PC World Communications Inc. The pace in the PC marketplace is too fast to rely on books alone.

## PC User Groups

Over the years, the single most important source of my PC information and knowledge comes from PC user groups. To steal a phrase from Tony the Tiger, *they're grrreat!*

If you aren't active in a PC user group, you really ought to give it a try. If you don't know how to find your nearest user group, look at Appendix I of *PC SECRETS,* published by IDG Books Worldwide, or contact the Association of PC User Groups (APCUG) for a list of user groups by writing them at 1730 M Street N. W., Suite 700, Washington D.C. 20036.

## On-Line Sources

If you have a modem, you can enjoy a wide variety of on-line information services that will extend and improve your DOS knowledge. Unless otherwise stated, you should set your communications software to no parity bits, 8 data bits, and 1 stop bit to access these services by modem.

To update the virus signatures to improve DOS 6's virus protection, call the Microsoft-sponsored BBS system at 1-503-531-8100. Appendix D of the *Microsoft MS-DOS User's Guide* explains the full procedure.

Microsoft also provides a general product-support BBS, known as the *Microsoft Product Support Download Service,* which contains text files that discuss DOS issues, as well as provides miscellaneous program updates. To access this BBS, call 1-206-936-6735.

Microsoft also runs a forum on CompuServe. To gain general access to the Microsoft forum, type `go microsoft` at a ! prompt. Alternatively, you can enter `go msdos6` to go directly to the DOS 6 section. CompuServe also provides a wealth of independent DOS information sources. Most major software vendors have their own forums that you can access by typing `go borland`, `go lotus`, and `go aldus`, for example. For more information on computer viruses, you can type `go virusforum` and for more information on the Association of Shareware Professionals, type `go shareware`.

If you are not currently a member of CompuServe and are interested in joining, you can call Microsoft to provide you with an introductory membership kit. To obtain the kit, call 1-800-848-8199 and, when prompted, press 1 and ask for operator 230.

A wealth of excellent BBSs are scattered all over the globe. If you do not know any BBS numbers, I suggest that you look at Appendix H of *PC SECRETS* or contact the Association of Shareware Professionals (ASP) for a list of ASP-approved bulletin boards. You can write to ASP at 545 Grover Rd., Muskegon, MI 40442.

## Information for Users with Disabilities

Microsoft is leading the way among the major software companies in providing additional support and services for users with disabilities. I hope that the other major vendors will follow Microsoft's lead.

For general information on Microsoft's options for users with disabilities, call 1-800-426-9400. If you use a TT or TDD compatible text telephone, call 1-206-635-4948.

AccessDOS extends DOS 6 to make it more useful to users with restricted dexterity by providing, for example, sticky Ctrl/Alt/Shift keys and mouse use via keyboard. The ADOS.COM program is available on the DOS 6 Supplemental disk; the program was developed by the Trace R&D Center at the University of Wisconsin-Madison.

For more information on AccessDOS and other products for PC users with disabilities, call Trace R&D Center using the voice line 1-608-263-2309 or the text telephone line at 1-608-263-5408, or write to them at the following address:

**Trace R&D Center**
S-151 Waisman Center
1500 Highland Avenue
Madison, WI 53705-2280

# Index

* * *

- 00n files, in backup, 83
- 4DOS, 42–43
- 486 microprocessor, 181
- 8086 microprocessor, 169, 174
  - IRQ assignments for, 244
  - memory management and, 186–187
- 80286 microprocessor, 178
  - IRQ assignments for, 245
  - memory management and, 186
  - Protected Mode, 178, 179
  - Real Mode, 178, 179
- 80386 microprocessor, 180
  - features, 180
  - memory management and, 186
  - running in Virtual 8086 Mode, 180
- ? switch, 14, 17, 229
  - UNINSTAL command and, 591
  - using, 44–45
- /? switch, 229
- /A switch, 2
  - DIR command with, 426–427
  - for displaying files, 53
  - for filemasks, 53
  - for listing subdirectories, 53
  - MSAV command with, 528
- /B switch
  - DEFRAG command with, 415
  - EDIT command with, 371
  - FC command with, 463
  - FIND command with, 470
  - FORMAT command with, 474
  - MEMMAKER command with, 507–508
- /BUFSIZE switch, 287
- /BW switch, 416
- /C switch
  - CHOICE command with, 309, 323
  - MEM command with, 187, 506
    - output from, 188–189
  - MSAV command with, 528
- /D switch, 20
  - MEM command with, 187, 506
  - MSCDEX command with, 533
  - SETVER command with, 566
  - XCOPY and, 47
- /DS switch, 94
- /DT switch, 94
- /E switch, 318
  - SETUP with, 812
  - XCOPY command with, 600
- /EDITOR switch, 312
- /F switch, 16
  - CHKDSK command with, 123
  - FORMAT command with, 473
  - MEM command with, 187
  - MSD command with, 61, 535, 536
  - SWITCHES command with, 265
- /FREE switch, 20
- /H switch, DOSKEY with, 438
- /I switch
  - FIND command with, 322
  - MIRROR command with, 515
- /INSERT switch, 287
- /K switch, 218
  - for COMMAND.COM, 334, 400
  - SWITCHES command with, 581
- /L switch, 20
  - DEVICEHIGH command with, 425
  - DIR command with, 428
  - FC command with, 462
  - LOADHIGH command with, 503
  - MSAV command with, 528
- /LIST switch, 145
  - DEVICEHIGH and LOADHIGH commands with, 194
- /M switch, 47
  - DOSKEY command with, 438
  - MEM command with, 187, 222, 330–331
  - SETUP with, 812
- /MOVE switch, 195
- /N switch, 16

- CHOICE command with, 323
- DEFRAG command with, 415
- DRIVPARM command with, 445
- MSAV command with, 529
- SWITCHES command with, 265

/O switch, DIR command with, 427

/O-D switch, 56

/P switch, 2
- COMMAND command with, 399
- EMM386 command with, 453
- MEM command with, 188, 505–506
- MSAV command with, 529
- MSD command with, 535
- PRINT command with, 549
- XCOPY command with, 47

/PARTIN switch, 71, 97

/Q switch, SETUP command with, 564, 565, 813

/QHELP switch, 19

/R switch, 104
- INTERSVR command with, 833

/REINSTALL switch, 287

/S switch, 2
- DEVICEHIGH command with, 425
- DIR command with, 427
- LOADHIGH command with, 503
- MSAV command with, 528
- MSCDEX command with, 533
- for shrinking memory allocation, 20
- with UNDELETE loaded, 91
- XCOPY command with, 47, 599

/T switch
- CHOICE command with, 309, 323
- MEMMAKER command with, 508
- with UNDELETE loaded, 91

/TEST switch, 97

/U switch
- DEFRAG command with, 141, 415
- FORMAT command with, 473
- for unloading UNDELETE, 91

/V switch, 291
- FORMAT command with, 473
- MSCDEX command with, 533

/W switch, 2
- SWITCHES command with, 580, 581

/X switch, 218
- INTERSVR command with, 494

**—A—**

A (DEBUG) command, 340–341

addressing
- of HMA, 182
- memory, 167–171
- Protected-Mode, 180

Advanced SCSI Programming Interface (ASPI), 256

ALARMCLK utility, 670

ALLSUB utility, 670

ANARKEY, 299–303, 613–618
- buffer management, 302
- command-line editing, 299–300
- for completing directory name, 299
- environmental editing, 300–301
- filenames and directories list, 300
- history lists, 301–302
  - illustrated, 301
- macros, 302
- MegaKey shortcut, 302–303
- MenuKey shortcut, 303
- pop-up environment window, 301
- registration form, 618
- *See also* DOSKEY

ANSI.SYS, 218–223
- for changing keystrokes, 237
- color display codes, 825–826
- cursor control codes, 227, 826–827
- description, 373
- determining if loaded, 222–223
- display attribute codes, 225, 826
- display codes, 224
- display mode control codes, 231, 827
- escape codes, 230
- escape sequences, 219
  - multiple, 221–222
  - special, 220
- example, 374
- going beyond, 239
- H command, 226

  keyboard control codes, 234–236, 828–830
  for line wrapping, 232
  options and switches, 373–374
  p command, 237
  quick reference, 825–830
  S command, 226
  sending instructions to, 218–222
    multiple escape sequences, 221–222
    using batch files, 219–220
    using PROMPT command, 220–221
    using TYPE command, 220
  switches supported by, 218
  text erasing codes, 826
  use of, 373
ANSISET.BAT, 227–229
  help screen, 229
APPEND command, 375–378
  examples, 376–377
  /PATH switches, 375, 377
  switches, 375–376
  usage, 375
  Windows and, 378
applications, help, 46
arenas, 200
ARJ, 618–631
  registration form, 631
Association of Shareware Professionals (ASP), 607
  members, 607
  ombudsman program, 608
  support, 608
ATTRIB command, 378–380
  archive attribute, 379
  examples, 379–380
  options, 378–379
  read-only attribute, 379
  to remove read-only flag, 418
  for removing hidden/system file attributes, 401
  switches, 379
  usage, 378
AUTOEXEC.BAT, 14
  checking, 281
  clean boot and, 262
  color codes and, 269
  commands, 281
  CONFIG environment variable and, 276–277
  configuring, 276–278
  examples, 22, 23
  interactive boot and, 263–264
  interrupting, 16
  MIRROR command and, 37
  synchronizing, with CONFIG.SYS, 15–16
  using MEMMAKER and, 197
    additional modifications, 203
    calling other programs and, 205
    IF statements and, 205
    optimization process modification, 202
  *See also* batch files; CONFIG.SYS

— B —

backup
  age of, 87
  convenience of making, 88
  copies of, 87
  differential, 79
  on DoubleSpace drives, 86
  file selection, 75–78
    using include/exclude clauses, 76–78
    using point and click, 76
  files created and used by, 81–83
  floppies, 83
    location of, 87
  full, 79
  incremental, 79
  making good, 80
  restoring files from, 84–85
  size limitation, 86
  speed optimization of, 81
  strategy for, 86–88
  tips, 73
  type selection, 78–79
BACKUP program, 74
  floppy disks and, 83
backup programs, 29–31
  operation limits, 86
  using, 74–86

- weaknesses of, 31
  - *See also* backup programs
- bank switching, 176
- base memory, 160
  - ANSICODE.BAT, improving, 223
  - BUFFERS statement, tuning, 209
  - environment size, minimizing, 208
  - FCBS allocation, minimizing, 210
  - FILES statement, tuning, 209
  - LASTDRIVE, minimizing, 210
  - MSCDEX consumption, minimizing, 210
  - saving, 207–210
  - STACKS command, optimizing, 209–210
  - trading, for extended memory, 211
  - *See also* memory
- batch files
  - adding efficiency and power to, 335–336
  - ANSIATTR.BAT, 225
  - ANSICODE.BAT, 219
  - ANSIOK.BAT, 222
  - ANSISET.BAT, 227–229
  - avoiding message prompts in, 314
  - BATUTIL and, 335–336
  - checking ERRORLEVELs in, 325
  - CHKFRAG.BAT, 316–317
  - command summary, 306–307
  - creating, 311
  - dates, times and, 323–328
  - for directory existence, 328–329
  - for DOS version check, 330
  - DOSKEY, 294
    - macros and, 294–295
  - editing, 311
  - explained, 306
  - function of, 306
  - ISDOS6.BAT, 330
  - moving beyond, 305–336
  - navigating around, 312–313
  - NEWPATH.BAT, 318–319
  - non-standard ASCII characters in, 311–312
  - OLDPATH.BAT, 318–319
  - ONLYONCE.BAT, 328
  - redirection and piping characters in, 313–317
  - routines, 328–332
  - running, once per day, 327–328
  - SAVEENV.BAT, 320
  - for saving current directory, 331–332
  - for sending instructions to ANSI.SYS, 219–220
  - SETDATE.BAT, 327
  - TCD.BAT, 331–332
  - TESTBOOT.BAT, 365
  - TESTFUNC.BAT, 362
  - TESTWILD.BAT, 329
  - TODAYIS.BAT, 324
  - tools for, 307
  - for TSR check, 330–331
  - TURBOBAT and, 336
  - WAITFOR.BAT, 326
  - for wildcard checking, 329–330
  - with Windows 3.1, 332–334
  - WINSTART.BAT, 333
  - *See also* AUTOEXEC.BAT; DOS, batch facility
- BATFILES directory, 377
- BATUTIL utility, 335–336, 386, 397, 632–645
  - character commands, 335–336
  - for cleaning up output, 447
  - for pausing utilities, 544
  - *See also* TURBOBAT utility
- binary files, 338
  - editing, 338–339
- BIOS, compression, 184
  - *See also* ROM-BIOS
- Bob's Better-Than-DOS Shareware Collection, 6–7, 610–611
  - ALARMCLK utility, 670
  - ALLSUB utility, 670
  - ANARKEY program, 299–303, 613–618
  - ANSISET.BAT batch file, 229
  - ARJ program, 618–631
  - BATUTIL utility, 335–336, 386, 397, 447, 544, 632–645
  - CALENDAR utility, 670–671
  - CLEANUP utility, 671–673
  - CMFiler utility, 66, 419, 421, 499, 585, 601, 645–659
  - contents list, 610–611

  DEBUG utility, 324
  Disk OrGanizer, 32, 141–142, 659–665
  DISKINFO utility, 673
  DUPE program, 433, 475, 666
  FANSI-Console, 239, 681–686
  FormatMaster, 475, 687–693
  HotDIR Plus, 57, 380, 429, 693–695
  LCD utility, 60, 388, 586
  Led's Change Directory, 696–697
  LIST program, 64, 526, 587, 698–707
  NO utility, 673–674
  Patriquin's Hard Disk Utilities, 708–732
  PCOPY utility, 48, 402, 601, 708–725
  PDAILY utility, 726
  PDEL program, 51
  PDELETE program, 97–98, 421, 726–732
  PocketD program, 48–49, 403, 421, 527, 601, 733–736
  PREFIX.EXE utility, 320–321
  QEDIT utility, 449, 737–751
  QFIND utility, 54, 172, 429, 752–759
  RELDEL utility, 51, 97–98, 419, 590, 759–760
  SCAN utility, 25
  SeekEasy utility, 469, 761–769
  SpinTest program, 127, 769–772
  SpinTime program, 127, 769–772
  STACKEY utility, 31, 41–42, 773–785
  TURBOBAT utility, 46, 336, 386, 396, 477, 544, 571
  TURBOTXT utility, 679–680
  UMB_DRV_SYS, 19–20, 424, 436, 456, 509, 785–800
  VIRUSCAN, 800–805
boot, system, 261
  determining, drive, 365
  initial, phase, 262
  interrupting and controlling, 261–265
    forcing [Y/N] prompt, 264
    SWITCHES command and, 264–265
    using clean-start key, 262–263
    using interactive start key, 263–264
  rebooting and, 361
BREAK command, 381–382
  Ctrl-Break and, 382
  examples, 381–382
  options, 381
  switches, 381
  usage, 381
buffer
  ANARKEY, 302
  command, 288
  DOSKEY, 286
  read-ahead, 383
buffering
  double, 131–132
    identifying, 574
  track, 130
  write, 130
BUFFERS command, 209, 382–384
  examples, 384
  options and switches, 383
  usage, 382
bulletin board services (BBSs), 607, 880–881
bus, 248
  address, 249
  AT, 251
  data, 249
  design, 248
  EISA, 252–255
  future of, 257
  innovations, 255–257
  ISA, 249–251
  local, 255–256
  mastering, 252
  MCA, 251–252
  role of, 249
  SCSI and, 256
  types, 249
    MSD option showing, 250
  VL, standard, 256

—C—

C (DEBUG) command, 341
caches. *See also* disk caches
CALENDAR utility, 670–671
CALL command, 307, 385–386
  examples, 386
  options and switches, 385
  usage, 385
  value of, 385

CAT files
    accessing, 84
    in backup, 83
    unable to find, 85
Catalog files, 82
CD command, 59, 386–388
    examples, 387–388
    options and switches, 387
    usage, 386
CD-ROMs
    accessing, 850
    MSCDEX, 210
    SMARTDRV and, 134
    *See also* MSCDEX command
CHCP command, 388–389
    examples, 389
    options and switches, 389
    usage, 388
CHDIR command, 59, 386–388
    examples, 387–388
    options and switches, 387
    usage, 386
checksum, 102
    getting rid of, files, 102
    scanning, 102
CHKDSK command, 123, 137, 390–393
    *.*, 392
    earlier versions of, 393
    examples, 391–392
    /F parameter and, 391
    options and switches, 390
    output generated by, 390–391
    running, 393
    usage, 390
CHKDSK tool, 28
    for determining file fragmentation on DoubleSpace drive, 126
    error messages, 124–126
    with /F switch, 123
    function of, 123–126
    using, 92
CHKLIST.CPS files, 101–102
CHOICE command, 35, 292, 393–396
    in batch files, 307
        example, 308
    with /C switch, 309, 323
    changing, 350–352
    disadvantages of, 308
    ERRORLEVEL and, 394–395
    examples, 395–396
    features, 308
    for fixed-period delay, 323
    for menu systems, 310
        illustrated, 211
        output, 310–311
    with /N switch, 323
    options and switches, 393–394
    pipe and filter characters and, 396
    slash mark and, 310
    with /T switch, 309, 323
    usage, 393
    valid keys display, 394
clean boot, 262
    DoubleSpace and, 263
    STACKER and, 263
    *See also* boot, system
clean-start key, 262–263
CLEANUP utility, 671–673
clock cycles, 167
clockspeed, 166
CLS command, 396–397
    for clearing screen and displaying message, 397
    example, 397
    options and switches, 396
    usage, 396
clusters, 115–116
    chains of, 120
    problems with SMARTDRV and, 134
    status of, 117
    *See also* sectors
CMFiler, 66, 419, 421, 499, 505, 590, 601, 645–659
    for advanced directory management tools, 556
    illustrated display, 66
    registration form, 659
    Tree Mode, 585–586
    for viewing files, 526

- CMOS chip, 165–166
- code-page management, 388–389
  - CHCP command and, 389
  - information files, 520
  - MODE device CP command and, 520
  - MODE SELECT command and, 389
- codes
  - ANSI.SYS, 825–830
  - break, 233
  - cursor control, 227
  - display, 224
  - display attribute, 225
  - display mode control, 231
  - keyboard control, 234–236
  - make, 233
  - PROMPT character, 551
  - scan, 232–233
    - understanding, 233
  - VSAFE option, 597
- colors
  - background, 224
    - code, 269
  - changing display, 218
    - *See also* ANSI.SYS
  - codes for MENUCOLOR command, 269, 510
  - EDIT, 312
  - foreground, 224
    - code, 269
  - low-intensity, 480
    - forcing, 552
  - suppressing, 480
  - for system prompt, 230
- COMMAND.COM, 334
  - resident block, 400
  - transient block, 400
- COMMAND command, 318, 397–400
  - for changing path temporarily, 400
  - description of, 397
  - examples, 400
  - /K switch with, 400
  - options and switches, 398–399
  - usage, 398
- command-line, 287
  - accessing, switches, 291–292
  - editing, 287–288
    - ANARKEY, 299–300
  - utilities, 846–849
    - NET USE, 847–848
    - NET VIEW, 846–847
    - reviewing directory examples and, 848
    - reviewing printer examples and, 849
- commands
  - ANSI, 223
  - assigning to keys, 237–239
  - AUTOEXEC.BAT, 281
  - batch file, 307
  - complex, 372
  - CONFIG.SYS, 279–280
  - configuration, 15
  - DEBUG, 339–346
    - entering, 346–349
  - directory navigation, 59–61
  - disabling, 296–297
  - discontinued DOS, 36
  - displaying syntax and usage of, 371
  - DOSKEY, 288–290
    - executing multiple, 290
    - saving, without executing, 289–290
    - selecting, by number, 288–289
    - using characters to select, 289
  - enhancing, 296–297
  - explanation of in *DOS 6 SECRETS*, 9
  - filenames and, 371–372
  - help sections of, 17
  - mandatory elements of, 371
  - menu, 15
  - network, 838
  - optional elements of, 371
  - with options and switches, 9
  - quick reference, 815–824
  - replacing, 296–297
  - startup, 16
    - combining, 269–272
  - submenu, 40
  - supplemental disk, 36
  - syntax for, 9
  - use conventions, 370–373
  - wildcards and, 372
  - Windows DOS session, to avoid, 334

commands, list of
  4HELP, 43
  @, 307
  ALIAS, 43
  ALLSUB, 51
  APPEND, 375–378
  ATTRIB, 378–380
  BACKUP, 380
  [BLOCKNAME], 15
  BREAK, 381–382
  BUFFERS, 382–384
  CALL, 307, 385–386
  CD, 59, 386–388
  CHCP, 388–389
  CHDIR, 59, 386–388
  CHKDSK, 123, 137, 390–393
  CHOICE, 35, 292, 307–310, 350–352, 393–396
  CHOICEAZ, 352
  CHOOSE, 278
  CLEANUP, 51
  CLS, 396–397
  COMMAND, 318, 397–400
  COPY, 47, 372, 400–403
  COPYTO, 298
  COUNTRY, 403–405
  CTTY, 405–406
  DATE, 406–407
  DBLSPACE, 27–28, 46, 145, 407–411
  DEBUG, 413–414
  DEFRAG, 31–32, 46, 292, 414–417
  DEL, 50, 153, 298, 417–419
  DELALL, 298
  DELOLDOS, 419–420
  DELTREE, 36, 52, 153, 420–421
  DEVICE, 422–423
  DEVICEHIGH, 20, 191, 194, 198, 423–425
  DHELP, 43
  DIR, 53–59, 208, 425–429
  DISKCOMP, 430–431
  DISKCOPY, 431–433
  DOS, 435–436
  DOSHELP, 45
  DOSKEY, 437–439
  DOSSHELL, 46, 380, 439–441
  DRIVPARM, 443–445
  ECHO, 218, 307, 314, 445–448
  EDIT, 46, 370–372, 373, 448–450
  EMM386, 451–456
  ERASE, 50
  EXIT, 456–457
  EXPAND, 457–458
  FASTHELP, 17, 45–46, 459–460
  FASTOPEN, 460–461
  FC, 461–464
  FCBS, 464–465
  FDISK, 465–467
  FILES, 467–468
  FIND, 54, 55, 322, 330, 468–470
  FOR, 307, 329, 470–472
  FORMAT, 290, 472–476
  GOTO, 307, 476–477
  GRAPHICS, 477–479
  HELP, 17, 40, 371, 479–481
  IF, 307, 328–329, 484–487
  INCLUDE, 15, 487–488
  INSTALL, 488–489
  INTERLNK, 489–491
  INTERSVR, 494–496
  KEYB, 496–498
  LABEL, 307, 498–499
  LASTDRIVE, 210, 499–500
  LOADFIX, 500–501
  LOADHIGH, 20, 191, 194, 198, 501–503
  MD, 504–505
  MEM, 20, 187–189, 222, 505–507
  MEMMAKER, 46, 507–509
  [MENU], 15
  MENUCOLOR, 15, 268, 509–511
  MENUDEFAULT, 15, 268, 511–512
  MENUITEM, 512–514
  MIRROR, 32, 37, 92, 514–515
  MODE command, 230–231, 516–524
    MODE COMn, 516–518
    MODE CON, 518–519
    MODE device CP, 519–521
    MODE device /STATUS, 521
    MODE display, 522–523
    MODE LPTn, 523–524
  MODE SELECT, 389

  MORE, 524–526
  MOVE, 36, 49–50, 526–528
  MSAV, 46, 528–530
  MSBACKUP, 29–30, 46, 530–532
  MSCDEX, 532–534
  MSD, 36, 54, 61–62, 172, 534–537
  MWAV, 537
  MWAVTSR, 537
  MWBACKUP, 538
  MWUNDEL, 46, 538–539
  NHELP, 43
  NLSFUNC, 539–540
  NO, 52
  NUMLOCK, 541
  PATH, 318, 369–370, 542–543
  PAUSE, 307, 543–544
  PCOPY, 48
  POWER, 544–546
  PRINT, 548–550
  PROMPT, 218, 220–221, 550–552
  QBASIC, 46, 552–553
  RD, 555–556
  REM, 307, 556–557
  RENAME, 315, 557–558
  REPLACE, 558–560
  RESTORE, 560–562
  SET, 208, 318, 563–564
  SETUP, 36, 564–565
  SETVER, 566–567
  SHARE, 568–569
  SHELL, 208, 569–570
  SHIFT, 307, 570–571
  SIZER, 571
  SMARTMON, 46
  SORT, 575–576
  STACKS, 209, 576–577
  SUBMENU, 15, 272, 578–579
  SUBST, 579–580
  SWITCH, 580–581
  SWITCHES, 16, 264–265
  SYS, 582–583
  TIME, 324–326, 583–584
  TREE, 584–586
  TYPE, 218, 220, 586–587
  UNDELETE, 31, 32–33, 587–590
  UNFORMAT, 31, 95–97, 592–593
  UNINSTAL, 147, 591–592
  VC, 291
  VER, 330, 593–594
  VERIFY, 594–595
  VOL, 595–596
  VSAFE, 596–598
  XCOPY, 17, 47–48, 598–601
  *See also* Workgroup Connection commands
Common Access Method Committee (CAM), 256
compression
  BIOS, 184
  data, 30, 143
  disk, 25–26
  of entire drive, 144–146
  floppy data, 156–157
  ratios, 155–156
    modifying, 155–156
  SMARTDRV and, 156
  *See also* DoubleSpace
CONFIG.SYS, 14
  boot menu file, 14–15
  BUFFERS statement, 209
  checking, 278–281
  clean boot and, 262
  configuration commands, 279–280
  confirmation before execution in, 17
  DBLSPACE.BIN and, 26–27
  DoubleSpace and, 148–149
  enhancements, 14
  entries to include in, 280
  examples, 22–23
  FCBS allocation in, 210
  FILES statement, 209
  increasing memory with, 189–192
  interactive boot and, 263–264
  interrupting, 16
  LASTDRIVE command in, 210, 500
  for loading ANSI.SYS, 218
  loading DBLSPACE.BIN high and, 195
  loading DOS high with, 189
  for loading INTERLNK.EXE, 493–494
  loading specific command in, 264

  MEMMAKER, 21
    additional modifications, 203
    MENU of options and, 204
    optimization process modification, 202
  menu with commands in COMMON section, 277–278
  menu with two options, 275–276
  with multiple configurations, 23
    startup, 265
  network entries in, 839–840
  STACKS command, 209–210
  for start-up menus, 266–267
    blockname, 267
    combining commands, 269–272
    command combination example, 271–278
    commands, 265–266
    commands, adding to beginning, 271
    defining, 266–267
    including [MEMSTUFF], 271
    simplifying, 270
  SWITCHES command and, 16
  synchronizing, with AUTOEXEC.BAT, 15–16
  for two-menu system, 273–274
  using MEMMAKER and, 197
  for utilizing UMBs, 191
  *See also* AUTOEXEC.BAT
configurations, 14–17
  AUTOEXEC.BAT, 276–278
  commands for, 15
  device driver switch and, 17
  installing minimum, 811–812
  interruption keys, 16
  IRQ, displaying, 246
  keeping track of, changes, 207
  locking up system with, 213–214
  MEMMAKER and, 204–205
  menus for, 14–16
    boot, 14
  multiboot, 265–278
  multimenu, 272–276
  multiple, 23
  power system, 261–282
controllers, 242
  interrupt, connecting two, 245
  SCSI, 112, 256
  twin-interrupt, 248
COPY command, 47, 372, 400–403
  /A switch, 372, 403
  /B switch, 372, 403
  for changing file date/time, 403
  examples, 402
  extended use, 403
  options and switches, 401
  usage, 401
COUNTRY command, 403–405
  examples, 404–405
  options and switches, 404
  usage, 404
creating
  ANARKEY macro, 302
  ASCII script files, 349
  batch files, 311
  directory, 306
  filemasks, 54–55
  multiboot configurations, 265–278
  prompts, 230
  utilities with DEBUG, 355–365
Ctrl-Alt-Del, system doesn't respond to, 214
Ctrl-Break, 382
CTTY command, 405–406
  examples, 406
  options and switches, 405
  with serial port, 406
  usage, 405
CURSFAT.COM program, 347
  listing, 347
  steps for creating, 348
CURSNORM.SCR, 349
cursor
  in batch files, 312–313
  control codes, 227
  positioning, 226
CVF (compressed volume) files, 144
  in full drive compression, 144–146
cylinder, 114
  breaking 1,024 barrier, 114
  *See also* tracks

## — D —

- D (DEBUG) command, 341–342
- data
  - bus, 249
  - compression, 143
    - floppy, 156–157
  - destruction, 97–98
  - encoding schemes, 113
  - recovering, 93–97
    - hard disk partition, 97
- data protection
  - precautions, 88–91
    - accidental file deletion, 88–89
    - installing deletion protection, 89
    - UNDELETE.INI and, 90–91
  - tools, 71–72
  - *See also* data
- data storage, 113
  - areas, 113
  - clusters, 115–116
  - file allocation table, 117
  - sectors, 114–115
  - tracks, 114
  - zone bit recording and, 115, 116
  - *See also* hard disks
- DATE command, 406–407
  - example, 407
  - options and switches, 406
  - usage, 406
- DBLSPACE.BIN, 26–27, 143, 148
  - CONFIG.SYS and, 27
  - loading, high, 195
- DBLSPACE command, 27–28, 407–411
  - /CHKDSK switch, 408
  - /COMPRESS switch, 410
  - for compression ratio modification, 155–156
  - /CREATE switch, 410
  - Custom Mode, 27
  - /CVF switch, 409
  - /DELETE switch, 409
  - examples, 411
  - Express Mode, 27
  - /FORMAT switch, 409
  - /INFO switch, 409–410
  - INI file, 411
  - /LIST switch with, 145
  - menu, 152
  - /MOUNT switch, 410
  - /NEWDRIVE switch, 408
  - /RATIO switch, 410
  - /RESERVE switch, 408
  - for resizing drive, 151
  - running, 410
  - /SIZE switch, 409
  - for summary of drives, 27–28
  - switches, 153
  - /UNMOUNT switch, 410
  - usage, 407–408
  - *See also* DoubleSpace
- DBLSPACE.SYS, 195, 412–413
  - example, 413
  - options and switches, 412
  - usage, 412
- DEBUG, 324, 337–366
  - ASCII script files with, 349
  - Assemble Mode, 349
  - changing cursor to block, 347
  - command reference, 339
  - commands, 339–346
    - ?, 339–340
    - A, 340–341
    - C, 341
    - D, 338, 341–342
    - E, 342
    - F, 342
    - G, 342
    - H, 342–343
    - I, 343
    - L, 343
    - M, 343
    - N, 344
    - O, 344
    - P, 344
    - Q, 339, 344
    - R, 344
    - S, 345
    - T, 345
    - U, 345

        W, 345
        XA, 346
        XD, 346
        XM, 346
        XS, 346
    commands, entering, 346–349
    CURSFAT.COM, 347–348
    date routines, 356–358
    editing with, 338–339
        files with EXE extension, 353
    features, 337
    help command, 340
    hex calculator, 354
    for looking for YN references, 351
    patching default EDIT filemask, 352–355
    patching programs with, 350–355
        changing CHOICE command, 350–352
        default EDIT filemask, 352–355
    segment alias support, 347
    utilities, creating, 355–365
        disk and drive, 363–365
        keyboard management, 358–362
        returning to DOS/setting ERRORLEVEL, 355–356
        working with dates, 356–358
    for viewing ASCII files, 339
DEBUG command, 413–414
    examples, 414
    options and switches, 413–414
    usage, 413–414
DEFRAG command, 31–32, 46, 292, 414–417
    /B switch, 415
    /BW switch, 416
    ERRORLEVELs from, 416–417
    examples, 417
    failure, 416
    /N switch, 415
    options and switches, 414–416
    /U switch, 415
    usage, 414
DEFRAG tool, 28, 31–32
    disk map, 32
    forcing into conventional memory, 292
    for keeping files unfragmented, 91
    memory problems, 139–142
    running, interactively, 138
    with /U switch, 141
    using, 137–139
defragmentation, 91
    benefits of, 136–142
    defined, 137
    of DoubleSpace volumes, 150–151
    of hidden files, 140
    methods, 138
        Files Only, 138, 140
        Full Optimization, 138, 139
    using DEFRAG for, 137–139
DEL command, 50, 153, 298, 417–419
    examples, 418
    options and switches, 417–418
    usage, 417
delete sentry, 88
    installing, 89
    using, 89
delete tracker, 88
    installing, 89
    using, 89
deleting, files, 50–52, 119–120
    *See also* undeleting
DELOLDOS command, 419–420
    example, 420
    options and switches, 419
    UNINSTALL and, 420
    usage, 419
DELTREE command, 36, 52, 153, 420–421
    ERRORLEVELs and, 421
    examples, 421
    options and switches, 420–421
    usage, 420
DEVICE command, 422–423
    example, 423
    options and switches, 422
    usage, 422
device drivers, 257
    .SYS extension, 472
    attributes table, 259
    block, 259
    categories of, 259
    character, 259
    CHKSTATE.SYS, 21

  COUNTRY.SYS, 404
  DBLSPACE.BIN, 26–27, 143, 148, 195
  DBLSPACE.SYS, 195, 412–413
  DEVICEHIGH installation, 424
  disk, 257
  DISPLAY.SYS, 433–434
  displaying, 258
  DoubleSpace, 152
    removing, 152–153
  DRIVER.SYS, 441–443
  EGA.SYS, 450
  EMM386, 451–456
  explained, 257–259
  help sections, 17
  HIMEM.SYS, 481–484
  installed with DEVICE, 423
  INTERLNK.EXE, 489, 491–494
  loading, 259
  POWER.EXE, 35, 546–548
  RAMDRIVE.SYS, 553–555
  routines, 257
  SETVER.EXE, 567–568
  SMARTDRV.EXE, 572–575
  switch for, 17
  *See also* ANSI.SYS
DEVICEHIGH command, 20, 191, 423–425
  example, 424
  extended use, 425
  /L switch, 425
  MEMMAKER and, 198
  options and switches, 423–424
  /S switch, 425
  usage, 423
  using, 194
dialog boxes
  Backup (from MSBACKUP), 74
  Backup (from MWBACKUP), 75
  Change Size, 151
  Edit Include/Exclude List, 76
    from MSBACKUP, 77
    from MWBACKUP, 78
  HELP Find, 41
  Select Backup Files, 76
  Select Catalog, 84
  Special Selections, 79
    from MWBACKUP, 80
DIR command, 53–59, 208, 425–429
  /A switch, 426
  displaying files, 53
    size of, 208
  examples, 428–429
  filemask creation and, 54–55
  /L switch, 428
  listing subdirectories and, 53
  /O switch, 427
  options and switches, 426–428
  /S switch, 427
  searching
    for directory name and, 55–56
    for files and, 54
  setting default, switches, 56–57
  Shareware alternatives to, 57–59
  usage, 425
DIRCMD variable, 5
  disabling, 56
directories, 118–119
  appended, 377
  batch file
    for existence of, 328–329
    for saving, 331–332
  changing, 298
  creating, 306
  file information in, 118
  moving up tree, 60
  navigation commands for, 59–61
  network examples, 848
  PATH command and, 542
  recovering, 96
  remote—accessing, 847–848
  renaming, 558
  in search paths, 318
  special file in, 119
  specifying, 76–77
disabled user information, 881
disk
  fragmentation, 31–32
  intercepting, changes, 297
  interweave, 126–128
    defined, 127
    illustrated, 127
    setting, 127–128

- partitioning, 121–123
- platter, 110
  - illustrated, 111
  - servoplatter and, 123
- read/write head, 110
  - illustrated, 111
- *See also* disk caches; disk utilities; hard disks

disk caches, 129
- assigning memory to, 129, 131
- bit rate, 129
- delayed write caching and, 132–133
- double buffering and, 131–132
- file mirroring and, 130
- functioning of, 130
- improving throughput with, 129–134
- performance, 33
- RAM drives and, 136
- status display, 132
- track buffering and, 130
- write buffering and, 130
- *See also* SMARTDRV program

Disk OrGanizer (DOG), 32, 659–665
- defragmentation methods, 142
- functions of, 141
- registration form, 665
- sample output from, 142
- switches, 141

disk utilities, 25–33
- disk fragmentation and, 31–32
- DoubleSpace disk compression, 25–26
- *See also* disk

DISKCOMP command, 430–431
- in batch files, 430
- ERRORLEVELs, 431
- examples, 431
- function of, 430
- options and switches, 430
- usage, 430

DISKCOPY command, 431–433
- ERRORLEVELs, 432
- examples, 432–433
- options and switches, 431–432
- usage, 431
- using, 432

DISKINFO utility, 673

display
- attribute codes, 225
- changing color, 218
- changing, lines, 230–232
- changing, modes, 230–232
- codes, 224
- jazzing up, 223–224
- mode control codes, 231
- setting, attributes, 225

DISPLAY.SYS, 433–434
- example, 434
- extended use, 434
- options and switches, 433–434
- usage, 433

DMA (direct memory access), 247
- backup speeds, 248
- concurrent read/write, 248
- dual, 248
- understanding, 247–248

DOS 6 SECRETS, 1–2
- audience for, 2
- commands explanation, 9
- conventions used in, 78
- disks, 6–7
- feedback on, 10
- organization of, 3–6

DOS 6, 1
- data protection tools, 71–72
- disadvantages, 13
- disk space required for, 809
- improvements from DOS 5, 2, 13
- installing, 809–813
- memory management tools, 185
- uninstalling, 813
- upgrading to, 1, 36–37, 809–810

DOS command, 435–436
- examples, 436
- HIMEM.SYS and, 435
- options and switches, 435
- usage, 435

DOS Resource Guide, 879

DOS Shell program, 62–68
- configuration details, 67
- file manager, 65

  - File menu, 66, 68
  - file selection/management, 64–66
  - file viewing with, 64
    - for mouse users, 65
  - Options menu, 64, 65, 66, 67
  - primary interface of, 63
  - task switching with, 67–68
  - uses, 64
  - View menu, 64, 65
- DOS
  - batch facility, 305
  - command interpreter, 400
  - extenders, 183–184
  - loading high, 189–190
- DOSHELP command. *See* FASTHELP
- DOSHELP.HLP, 45
  - custom version of, 45
- DOSKEY
  - identifier, 292
  - buffer size, setting, 286
  - changing default mode, 286
  - command history, 288
    - erasing, 290
    - listing, 289
  - as command-line manager, 287–290
  - commands, 288–290
    - characters to select, 289
    - entering multiple, 293
    - executing multiple, 290
    - saving, without executing, 289–290
    - selecting, by number, 288–289
  - functions, 285
  - going beyond, 299–303
  - Insert Mode, 286
  - installing, 285–286
    - in Insert Mode, 286
  - keys for command-line editing, 438
  - keys, most often used, 288
  - macros, 290–294
    - accessing command-line switches and, 291–292
    - batch file comparison, 294–295
    - common task, 298
    - extending search path, 298
    - installed, 293
    - intercepting drive changes, 297
    - question-mark help, 295
    - reinstalling, 291
    - removing, 291
    - replacing/disabling DOS commands, 296–297
    - reviewing, settings, 293
    - supporting multiple file names, 297–298
    - tricks, 295–298
  - Overtype Mode, 286
  - piping with, 292–293
  - redirection with, 292–293
  - reinstalling, 287
  - *See also* ANARKEY
- DOSKEY command, 437–439
  - /BUFSIZE switch, 437
  - examples, 438–439
  - for executing simultaneous multiple commands, 439
  - /H switch, 438
  - /M switch, 438
  - options and switches, 437–438
  - /REINSTALL switch, 437
  - usage, 437
  - *See also* DOSKEY
- DOSSHELL command, 46, 380, 439–441
  - to change file attributes, 380
  - examples, 441
  - Graphics Mode, 440
  - options and switches, 440
  - for renaming directories, 558
  - Text Mode, 440
  - usage, 439–440
- DOSSHELL.INI, 67
- DoubleSpace, 25–26
  - algorithms, 143
  - booting process, 148–149
  - CHKDSK and, 126
  - clean boot technique and, 263
  - compressing floppies with, 156–157
  - defragmenting, volumes, 150–151
  - device driver, 26, 27
  - disadvantages of, 28
  - drives, 144–146
    - compression ratios, 155–156
    - custom setup, 146–147

    - express option, 146
    - host letter, 154
    - installing, 146–151
    - letter assignments, 153–155
    - removing, 152
    - resizing, 151–152
    - size limitations, 147
    - Stacker and, 147
  - error messages, 150
  - features, 142
  - files, 148
    - INI, 148
  - full-screen illustrated, 29
  - function of, 142–143
  - mechanics, 143–146
  - with SMARTDRV, 133, 134
  - swap files and, 149–150
  - using, 142–157
  - *See also* DBLSPACE command
- DOUBLESPACE.SYS, 27
- DPMI (DOS Protected Mode Interface), 183
- DR DOS, 1
- DRAM chip, 163–164
  - illustrated, 163
  - parity value and, 163
  - *See also* SRAM chip: RAM
- DRIVER.SYS, 441–443
  - examples, 442–443
  - extended use, 443
  - factor values, 442
  - options and switches, 441–442
  - usage, 441
  - using, 442
- DRIVPARM, 443–445
  - factor values, 444
  - /N switch, 445
  - options and switches, 444–445
  - usage, 443
- DUPE program, 433, 475, 666
  - registration form, 666

— E —

- E (DEBUG) command, 342
- ECHO command, 218, 307, 314, 445–448
  - in date/time batch files, 323–328
  - for displaying blank line, 447–448
  - examples, 446–447
  - functioning of, 447
  - options and switches, 446
  - usage, 445–446
  - uses of, 446
- EDIT, 311
  - colors, 312
  - for creating ASCII file, 349
  - /EDITOR switch and, 312
  - features not supported by, 313
  - filemask, default, 352–355
  - launching, 298
  - QBASIC rather than, 312
- EDIT command, 370, 448–450
  - /B switch in, 371
  - for CGA systems, 448
  - on EGA systems, 449
  - examples, 449
  - with filename variations, 371–372
  - functions of, 449
  - for LCD systems, 448
  - options and switches, 448–449
  - syntax, 373
  - usage, 448
- editing
  - ANARKEY
    - command-line, 299–300
    - environmental, 300–301
  - batch files, 311
  - command-line, 287–288
    - DOSKEY, 438
  - with DEBUG, 338–339
  - DOSKEY keys, 287–288
- EEMS (Enhanced Expanded Memory Specification), 175
- EGA.SYS, 450
  - example, 450
  - options and switches, 450
  - usage, 450
- EISA bus, 252–255
  - illustrated, 253, 254
  - ISA bus comparison, 253, 255
  - MCA bus comparison, 253
- EMM386, 23, 451–456
  - 286 systems and, 186
  - Auto Mode, 452

  D switch, 248
  enhancements, 23
  examples, 456
  FRAME= switch, 453
  function of, 192
  HIGHSCAN option, 454
  MEMMAKER and, 455
  NOEMS switch, 452
    added to, 214
    using, 454
  NOHI option, 454
  options and switches, 451–455
  /P switch, 453
  usage, 451
  Weitek coprocessor and, 452
EMS (expanded memory specification), 176
  history of, 175
  page frame, 176
  *See also* expanded memory
environment, DOS, 208
  assigning new data to, variable, 318
  changing search path and, 318–319
  increasing, 318
  innovations, 317–322
  minimizing size of, 208
  PROMPT variable, 318
  restoring, 320–321
  saving, 320–321
  storing date in, variable, 327
  TEMP variable, 320
  uppercasing input in, 321–322
ERASE command. *See* DEL command
Error Correction Code (ECC), 81
error messages
  CHKDSK, 124–126
  DoubleSpace, 150
ERRORLEVELs, 320
  batch files, checking, 325
  from DEFRAG failure, 416–417
  DISKCOMP, 431
  DISKCOPY, 432
  FORMAT, 474
  IF command, 485, 486
  for indicating if search was found, 330
  KEYB, 497–498
  REPLACE, 559–560
  RESTORE, 562
  set by KEYSTATE.COM, 359
  setting, 355–356
  support, 322
  WEEKDAY.COM and, 324
  WHATYEAR.COM and, 324
  XCOPY, 600
ESDI (Enhanced Small Device Interface), 112
EXIT command, 456–457
  example, 457
  options and switches, 457
  usage, 457
EXPAND command, 457–458
  examples, 458
  options and switches, 458
  usage, 457
expanded memory, 23, 160
  defined, 176
  managing, 211–212
  *See also* EMS (expanded memory specification)
Extended Industry Standard Architecture. *See* EISA bus
extended memory, 160, 176–182
  defined, 178
  explained, 178–179
  from free base memory, 211
  illustrated, 177
  maximizing, 210–211
  *See also* memory
ExtraDOS Toolbox Professional, 51, 667–680
  ALARMCLK, 670
  ALLSUB, 670
  CALENDAR, 670–671
  CLEANUP, 671–673
  DISKINFO, 673
  NO, 673–674
  registration form, 680
  tools list, 51–52
  TURBOBAT, 46, 336, 386, 396, 477, 487, 571, 674–679
  TURBOTXT, 679–680

## — F —

F (DEBUG) command, 342
FANSI-Console, 239, 681–686
  features, 239
  registration form, 686
Fastdisk, 135
  using, 135
FASTHELP command, 17, 459–460
  customizing, 46
  examples, 459
  options and switches, 459
  output printout, 460
  usage, 459
  using, 45–46
FASTOPEN command, 460–461
  disadvantages, 461
  example, 461
  options and switches, 460
  usage, 460
FASTOPEN utility, 136
FC command, 461–464
  /B switch, 463
  examples, 463–464
  /L switch, 462
  options and switches, 462–463
  usage, 461–462
FCBS command, 464–465
  example, 465
  options and switches, 465
  usage, 465
FDISK command, 465–467
  examples, 466
  options and switches, 465–466
  usage, 465
  warning, 467
FDISK program, 122
FHELP batch file, 42
file allocation table (FAT), 117
  12-bit, 118
  16-bit, 118
  cluster status, 117
  in deleting files, 119
  partitioning and, 122
  in reading files, 119
  in writing files, 120
  *See also* files
filemasks
  /A switch for, 53
  creating, 54–55
  EDIT, default, 352–355
  multiple, commands for, 297–298
  specifying, 76–77
  *See also* Files
filenames
  entering, 371–372
  supporting multiple, 297–298
files
  ASCII script, 349
  backup selection of, 75–78
    options for, 79
    using include/exclude clauses, 76
    using point and click, 76
  batch, 219–220
  compressed volume, 144
  compression of, 143–144
  configuration, 14
  created and used by backup, 81–83
    00n, 83
    CAT, 83
    Catalog, 82
    INI, 83
    LOG, 83
    SET, 82
    SLT, 82
  decompression of, 143–144
  DEFRAG for unfragmenting, 91
  defragmenting, 137–142
    hidden, 140
  deleting, 50–52, 119–120
    all, 51
  deletion of, accidental, 88–89
  displaying, 53
  DoubleSpace, 86, 148
  erasing, 97–98
  fragmented, 31, 136–137
    determining, 137
    ialso defragmentation
  reading, 119
  recovering, 32–33

lost, 93–97
with UNFORMAT, 97
restoring, from backup, 84–85
searching for, 54
selected, identification of, 76
selection/management of, 64–66
swap, 149–150
undeleting, 93–95
viewing in DOS Shell, 64
writing, 120
*See also* file allocation table (FAT); filemasks; filenames
FILES command, 209, 467–468
example, 467
options and switches, 467
usage, 467
FIND command, 54, 55, 468–470
/B switch, 470
enhancements, 322
ERRORLEVELs and, 330
examples, 469
/I switch with, 322
no filename specification, 469
options and switches, 468
usage, 468
FINDDIR.BAT, 55
contents of, 55
example of using, 56
FLEXDIR.BAT, 55
FOR command, 307, 329, 470–472
examples, 471–472
options and switches, 470–471
usage, 470
FORMAT command, 290, 472–476
/B switch, 474
ERRORLEVELs, 474
examples, 475
/F switch, 473
options and switches, 472–474
/U switch, 473
undocumented switches, 475–476
usage, 472
/V switch, 473
FormatMaster program, 475, 687–693
registration form, 693
formatting
high-level, 120, 121
low-level, 120–121
destructive, 128
*See also* unformatting

— G —

G (DEBUG) command, 342
GOTO command, 307, 476–477
example, 476–477
options and switches, 476
usage, 476
GRAPHICS command, 477–479
example, 479
/LCD switch, 479
options and switches, 477–479
supported printer types, 478
usage, 477

— H —

H (DEBUG) command, 342–343
hard disks, 109
alignment problems, 128
checking characteristics of, 116
components, 110
controllers, 112
explained, 112–113
DoubleSpace letter assignments, 153–155
illustrated, 111
inside, 110–111
mechanics of, 110–113
MSD program and, 116–117
partitioning, 121–123
reasons for, 122
preparing new, 120–123
disk partitioning, 121–123
low-level formatting, 120–121
saving onto, 113
speeding up, 126–142
surface deterioration, 128
*See also* data storage

Help
  applications, 46
  index, 19
  invoking, system, 43
  irritations of, 19
  menu, 18
  notes example, 18
  on-line, 17–19
  question-mark, 295
  searches, 40–42
  tricks for Microsoft C programmers, 43
HELP command, 17, 479–481
  4DOS/NDOS and, 42–43
  examples, 481
  Find dialog box, 41
  help text for, 43
  options and switches, 480
  submenu commands, 40
  subtopics, 481
  suppressing colors with, 480
  for syntax and usage, 371
  usage, 480
HELP.HLP, 19, 43
hexadecimal numbers, 167–169
high memory. *See* upper memory
HIMEM.SYS, 189, 481–484
  effects of using, 190
  example, 484
  host computer table, 483
  installation of, 484
  /INT15 switch, 482
  /MACHINE switch, 483
  /NUMHANDLES switch, 483
  options and switches, 482–484
  /SHADOWRAM switches, 484
  slowing of performance, 482
  usage, 481
HMA (high memory area), 182
  addressing of, 182
  using, 482
  *See also* upper memory
HotDIR Plus, 57, 380, 429, 693–695
  display, 57
  registration form, 693–695

—I—

I (DEBUG) command, 343
IDE (Integrated Drive Electronics)
    controller, 112
IF command, 307, 328–329, 484–487
  ERRORLEVELs, 485, 486
  examples, 486
  NOT operand, 485
  options and switches, 485
  usage, 484–485
INCLUDE command, 15, 487–488
  example, 488
  options and switches, 487
  usage, 487
  using, 487–488
include/exclude clauses, 76–78
Industry Standard Architecture. *See* ISA bus
INI files
  in backup, 83
  Windows data protection utilities and, 73
  *See also* DOSSHELL.INI files
INSTALL command, 488–489
  example, 489
  memory-resident programs and, 489
  options and switches, 489
  usage, 488
installing
  DOS 6 SECRETS disks, 609
  DOS 6, 809–813
    manually, 812
  DOSKEY, 285–286
  to floppies, 810–811
  INTERLNK, 832–833
  INTERSVR, 833
  Microsoft Mail, 850–851
  minimum configuration, 811–812
  network software, 836–838
  to nonboot drive, 812–813
  uninstalling DOS 6 and, 813
  Windows components, 812
InterLink, 831–835
  establishing, connection, 832–834
  limitations, 835

  speed of, 835
  steps to successful linking, 833–834
  using server resources and, 834
  wiring computers together and, 831–832
INTERLNK command, 489–491
  description of, 489
  examples, 490
  functions of, 489
  options and switches, 490
  parallel cable for use with, 491
  usage, 490
INTERLNK.EXE, 489, 491–494, 832–833
  /COM switch, 492
  defaults, 492–493
  /DRIVES switch, 832
  examples, 493–494
  loading, 493–494
    UMBs and, 492
  /LPT switch, 492
  options and switches, 492–493
  usage, 491
INTERLNK program, 35, 835
  installing, 832–833
interrupt
  controllers, connecting two, 245
  handler, 243
  request lines, 244–245
    *See also* IRQs
  service routine (ISR), 243
  vector table, 243–244
  vectors, 243–244
  *See also* interrupts
interruption keys, 16
interrupts, 242
  categories of, 242
  hardware, 242
  software, 242
  *See also* interrupt, vectors
INTERSVR command, 494–496
  /COM switch, 494
  examples, 496
  extended use, 496
  /LPT switch, 495
  multitasking and, 495
  options and switches, 494–495
  /R switch, 833
  usage, 494
  /X switch, 494
INTERSVR program, 835
  installing, 833
interweaving
  disk, 126–128
  memory, 167
IRQs, 244
  assignments
    for 8086/8088-based systems, 244
    for 80286-based systems and higher, 245
    reviewing, 246–247
  displaying configuration, 246
  hardware devices and, 247
ISA bus, 249–251
  EISA bus comparison, 253, 255
  illustrated, 254
  MCA bus comparison, 252

## —K—

KEYB command, 496–498
  for Enhanced Keyboard, 497
  ERRORLEVELs, 497–498
  example, 498
  options and switches, 497
  usage, 497
keyboard
  control codes, 234–236
  customizing, 232–239
  management utilities, 358–362
    changing state of special keys, 358–360
    rebooting PC, 361
    waiting for function key, 361–362
  scan codes and, 233
keys
  assigning commands/strings to, 237–239
  checking/changing state of, 358–360
  clean-start, 262–263
  codes for, 232–237
  DOSKEY, 288
    command-line editing, 438

  editing, DOSKEY, 287–288
  interactive start, 263–264
  interruption, 16
  remapping, 237
  start, controlling, 264–265
KEYSTATE.COM, 358
  ERRORLEVELs set by, 359

—L—

L (DEBUG) command, 343
LABEL command, 307, 498–499
  examples, 499
  options and switches, 498–499
  on SUBST drive, 499
  usage, 498
laptop tools, 35
LASTDRIVE command, 210, 499–500
  example, 500
  options and switches, 500
  usage, 500
LCD utility, 60–61, 388, 586
  illustrated display, 60
Led's Change Directory, 696–697
  registration form, 697
LH command. *See* LASTDRIVE command
line wrapping, 232
LIST program, 64, 526, 587, 698–707
  registration form, 707
LOADFIX command, 500–501
  example, 501
  options and switches, 501
  usage, 501
LOADHIGH command, 20, 191, 501–503
  example, 502
  extended use, 503
  /L switch, 503
  MEMMAKER and, 198
  options and switches, 502
  programs available to use, 502
  /S switch, 503
  system lockup and, 214
  usage, 502
  using, 194

local bus, 255–256
  connectors, 256
lockup, system
  with new configuration, 213–214
  program causes, 214
  while running MEMMAKER, 212
LOG files, in backup, 83

—M—

M (DEBUG) command, 343
macros
  ANARKEY, 302
  DOSKEY, 290–294
magazines and periodicals, 879–880
MAIL command, 850, 856–858
  -C switch, 856
  -D switch, 856
  examples, 858
  -N switch, 857
  options and switches, 856–857
  setup confirmation list, 851
  usage, 856
  -X switch, 857
  *See also* Microsoft Mail
master partition table, 122, 466
  active partition, 466
  function of, 123
  *See also* partitioning, disk; partitions
MCA bus, 248, 251–252
  EISA bus comparison, 253
  illustrated, 255
  ISA compared to, 252
MD command, 504–505
  examples, 504–505
  options and switches, 504
  usage, 504
MEM commands, 20, 187–189, 505–507
  /C switch, 506
  /D command, 257, 506
    output listing, 258
  example output, 187
  examples, 506
  /M switch, 187, 222, 330–331
  options and switches, 505–506

  /P switch, 505–506
  switches used with, 187–188
  UMBs and, 191–192
  usage, 505
MEMMAKER command, 46, 507–509
  /B switch, 507–508
  examples, 508–509
  extended use, 509
  options and switches, 507–508
  for suppressing confirmation prompts, 509
  /T switch, 508
  /UNDO switch, 508
  usage, 507
MEMMAKER program, 20–23
  advance options, 21
  AUTOEXEC.BAT modification, 202
    additional, 203
    calling programs and, 205
    IF statements and, 205
  CHKSTATE.SYS utility, 21, 201
  CONFIG.SYS modification, 202
    additional, 203
    MENU of options and, 204
  Custom Mode, 198, 199, 508
    configuration options, 200
  DOS= statements, 191
  DOS=HIGH statement, 198
  example of using, 196–197
  excluding programs from analysis, 206
  explained, 201–204
  Express Mode, 198, 199, 508
  function of, 195
  INI file, 206
  locking up of system with, 212
  main option screen, 198, 199
  modes of running, 20–21
  multiple configurations and, 205
  NOEMS switch, 199
  running, 198–201
  SHELL command and, 570
  SIZER.EXE utility, 21, 201, 571
  success of, 204–206
memory
  addressing, 167–171
    hexadecimal numbers, 167–169
    multiple, 170
    segments and offsets, 169–171
  assigning to cache, 129, 131
  base, 160
    saving, 207–210
  cache, 165
  checking, 171–172
  conflicts, solving, 212–214
  conventional, 160
  DEFRAG problems, 139–142
  EMM386 enhancements and, 23
  expanded, 23, 160, 175–176
    managing, 211–212
  extended, 160, 176–182
    maximizing, 210–211
  hardware, 162–167
    CMOS chip, 165–166
    DRAM chip, 163–164
    SIMMs, 165
    speed issues, 166–167
    SRAM chip, 164–165
  high, 182
  interleaved, 167
  management, 19–23
    for 8086 users, 186–187
    for 80286 users, 186
    for 80386 users, 186
    tools, 185
  managers, 183
  maximizing, 159–215
  optimization tools, 20–23
  optimizing, 167
  paging, 176
  parity errors, 164
  random-access (RAM), 161
  read-only (ROM), 161
  removing DoubleSpace device driver from, 152–153
  ROM-BIOS (read-only memory-basic input/output system), 162
  speed issues, 166–167
    clock cycles and, 167
    clock speed and, 167
    precharge time and, 167
    wait state and, 167

- terms, 161
- types, 171–182
- upper, 131–132, 172–175, 189–190, 193, 195, 200–201, 204
  - blocks, *See* UMBs (upper memory blocks)
- video, 174
- virtual, 184–185
- warehouse, 174–175
  - illustrated, 174

MENUCOLOR command, 15, 268, 509–511
- color codes, 510
- color codes for, 269
- examples, 510–511
- options and switches, 510
- placement of, 510
- usage, 509

MENUDEFAULT command, 15, 268, 511–512
- example, 512
- options and switches, 511
- specifications, 512
- usage, 511

MENUITEM command, 512–514
- blocknames, 513
- examples, 513
- options and switches, 512–513
- usage, 512

menus
- adding programs to, 67–68
- configuration, 14–16
- Help, 18
- multimenu configurations and, 272–276
  - escape option, 274
  - user confirmation prompting, 275–276
- new commands for, 15
- *See also* start-up menus

messages, suppressing, 314–316
- *See also* error messages

MFM (modified frequency modulation), 113

Micro Channel Architecture. *See* MCA bus

MICRO command, 853, 858–860
- examples, 860
- mailbox creating before running, 859
- options and switches, 858–859
- sample, message, 854
- switches, 853
- usage, 858

microprocessors, 242
- 486, 181
- 80286, 178–179
- 80386, 180–181
- DX/DX2, 181
- interrupting, 242
- SX, 181
- *See also* Environment and Tools manual, 43

Microsoft Mail, 850
- features, 852
- files, locating, 852–853
- installing, 850–852
- mailbox name, 855
- password, 855
- postoffice, 858
- receiving new mail with, 853–855
- running, 852
- -X switch, 853
- *See also* MAIL command

Microsoft telephone support, 879

MIRROR command, 32, 37, 92, 514–515
- description of, 514
- examples, 515
- /I switch, 515
- options and switches, 514–515
- usage, 514

MIRROR program, 71
- features missed by, 92
- UNFORMAT and, 96–97

MKDIR command. *See* MD command

MODE command, 230
- control codes, 231
- for setting number of characters per line, 231
- syntax, 230
- *See also* MODE subcommands

MODE COMn command, 516–518
- examples, 518
- options and switches, 517–518
- supported transmission rates, 517
- transmission errors and, 518
- usage, 516

MODE CON command, 518–519
  example, 519
  options and switches, 519
  usage, 518
MODE device CP command, 519–521
  code-page information files, 520
  example, 521
  options and switches, 520
  SELECT switch, 520
  supported devices, 520
  usage, 519
MODE device /STATUS command, 521
  examples, 521
  options and switches, 521
  usage, 521
MODE display command, 522–523
  display values, 522
  examples, 523
  formats, 522
  options and switches, 522
  for restoring cursor, 523
  usage, 522
MODE LPTn command, 523–524
  example, 524
  extended use, 524
  options and switches, 523
  retry setting values, 524
  usage, 523
MODE SELECT command, 389
MORE command, 524–526
  examples, 525
  options and switches, 525
  usage, 525
MOVE command, 36, 49–50, 526–528
  COPY command differences, 50
  examples, 527
  options and switches, 526
  source/target requirement, 527
  syntax, 49
  usage, 526
  using, 49–50
MOVE'EM, 186
MSAV command, 528–530
  /A switch, 528
  /C switch, 528
  CHKLIST.MS and, 529–530
  examples, 530
  /L switch, 528
  /N switch, 529
  optional settings, 530
  options and switches, 528–529
  /P switch, 529
  /S switch, 528
  usage, 528
MSAV program, 24–25, 71
  illustrated, 24
  INI file, 102
  MWAV comparison, 104
  updating signature lists, 101
  uses, 98
MSBACKUP command, 29–30, 46, 530–532
  backup settings, 531
  examples, 532
  options and switches, 530–531
  SET file searches, 531
  usage, 530
MSBACKUP program, 29, 46, 71
  backup speed, 248
  catalog menu options, 85
  Compare option, 80
  dialog box, 30
  ECC system, 81
  features, 29–30
  MWBACKUP comparison, 74–75
  STACKEY utility and, 31
MSCDEX command, 532–534
  /D switch, 533
  examples, 534
  loading, 533–534
  options and switches, 533
  /S switch, 533
  usage, 532
  /V switch, 533
  *See also* CD-ROMs
MSCDEX utility, 210
MSD command, 36, 54, 61–62, 172, 534–537
  examples, 537
  /F switch, 535, 536
  /filename switch, 536
  options and switches, 535

  /P switch, 535
  usage, 535
MSD program, 36, 54, 61–62
  for checking memory, 171–172
  for determining UART type, 133
  Disk Drives window, 117
  for displaying device drivers, 258
  for displaying IRQ configuration, 246
  file finding/browsing tool, 62
  for hard disk info, 116
  illustrated display, 62
  for loaded locations in upper memory, 204
  memory/computer displays for, 173
  for showing bus type, 250
MSDOSDATA variable, 73
multimenu configurations, 272–276
  escape option, 274
  user confirmation, 275–276
  *See also* start-up menus
MWAV command. i MSAV command
MWAV utility, 33–34, 46, 72
  illustrated, 100
  INI file, 102
  MSAV comparison, 104
  updating signature lists, 101
  uses, 98
  *See also* MSAV command; MSAV program
MWAVTSR command, 537
MWBACKUP command, 538
  *See also* MSBACKUP command
MWBACKUP program, 29, 33–34, 46, 72
  backup speed, 248
  Compare option, 80
  ECC system, 81
  features, 29–30
  MSBACKUP comparison, 74–75
  for recovering files, 33
  running, 81
  *See also* MSBACKUP program
MWUNDEL command, 46, 538–539
  options and switches, 538
  undeletion success value, 539
  usage, 538
MWUNDEL utility, 33–34, 46, 72
  deleted files displayed by, 94
  using, 93

— N —

N (DEBUG) command, 344
NDOS, 42–43
NET command, 838, 860–861
  description, 860–861
  subcommands, 860–861
  summary, 839
NET CONFIG command, 861–862
  example, 862
  options and switches, 861
  usage, 861
NET LOGOFF command, 862–863
  example, 863
  options and switches, 862
  usage, 862
NET LOGON command, 842, 863–864
  examples, 864
  options and switches, 863
  usage, 863
  username.PWL file, 864
  /YES switch, 863
NET PASSWORD command, 843, 865–866
  examples, 865
  extended use, 866
  options and switches, 865
  usage, 865
NET PRINT command, 849, 866–868
  examples, 867–868
  functions of, 867
  options and switches, 866–867
  usage, 866
NET START command, 841, 868–870
  examples, 869
  functions, 842
  NetBEUL and, 869
  NetBIOS and, 869
  options and switches, 868–869
  usage, 868
NET STOP command, 870–871
  common use of, 871
  examples, 871

  options and switches, 870
  usage, 870
NET TIME command, 871–872
  example, 872
  options and switches, 871–872
  usage, 871
NET USE command, 847, 872–875
  ? character and, 873
  /DELETE switch, 873
  examples, 874–875
  options and switches, 872–874
  /PERSISTENT switch, 874
  syntax, 848
  usage, 872
NET VER command, 876
  options and switches, 876
  usage, 876
NET VIEW command, 846–847, 876–877
  examples, 877
  options and switches, 876–877
  usage, 876
NETBEUI protocol, 841
  description of, 869
NETBIOS protocol, 841
  description of, 869
network
  commands, 838
  CONFIG.SYS entries, 839–840
  connecting to, 838–843
  how to log on, 841–842
  protocols, 841
  resources, using, 843–850
    CD-ROM access, 850
    command-line utilities, 846–849
    Pop-Up utility, 843–846
  software, 836
    installing, 836–838
NLSFUNC command, 539–540
  example, 540
  options and switches, 540
  usage, 540
NO utility, 673–674
Norton
  Desktop for Windows (NDW), 72
  Disk doctor, 92
NUMLOCK command, 541
  example, 541
  options and switches, 541
  usage, 541

— O —

O (DEBUG) command, 344
on-line
  help, 17–19
  sources, 880–881

— P —

P (DEBUG) command, 344
parity
  checks, 164
  errors, 164
  value, 163
partitioning, disk, 121–123
  active partition and, 123
  FDISK program for, 122
  master partition table and, 122
  reasons for, 122
partitions, 466
  active, 466
  types of, 466
password
  management, 842–843
  Microsoft Mail, 855
PATH command, 318, 369–370, 542–543
  character limits, 543
  examples, 543
  improving performance and, 542
  options and switches, 542
  usage, 542
paths
  changing, temporarily, 318–319, 400
  searching, 61
Patriquin's Hard Disk Utilities, 708–732
  PCOPY utility, 708–725
  PDAILY utility, 726
  PDELETE utility, 726–732
  registration form, 732

PAUSE command, 307, 543–544
  CHOICE command and, 544
  example, 544
  options and switches, 543
  usage, 543
  PC user groups, 880
  PCI (Peripheral Component Interconnect), 256
PCOPY utility, 48, 402, 601, 708–725
  advantages of, 73
  illustrated, 48
  /X switch to, 50
PDAILY utility, 726
PDEL program, 51
PDELETE utility, 97, 421, 726–732
  using, 98
PIC (programmable interrupt controller), 244
pipe characters, 292, 313
  substitutions for, 293
piping
  for avoiding message prompts, 314
  command output to FIND, 317
  DIR command output to FIND, 469
  with DOSKEY, 292–293
  ECHO output to command, 314
  examples, 314–315
  *See also* redirection
PKZIP program, 143, 144
PocketD program, 48–49, 403, 421, 469, 527–528, 601, 733–736
  deleting files with, 51
  as DIR alternative, 58
  file size analysis, 58
  file size analyzation, illustrated, 59
  flexible wildcards and, 58
  registration form, 736
  using Tiny Mode, 58
Pop-Up program, 838
  Browse option, 845–846
  connecting to directories, 845–846
  connecting to shared printers, 846
  Disk Connections display, 844
  Form Connections dialog box, 846
  loading and running, 843–845
  in monochrome systems, 844
  removing TSR, 844
  using, 843–846
POWER command, 544–546
  APM hardware systems and, 545
  examples, 546
  MAX switch, 545
  options and switches, 545
  STD switch, 546
  usage, 545
POWER.EXE, 35, 546–548
  APM hardware systems and, 547
  examples, 547
  loading, 547
  location of, 546
  MAX switch, 547
  options and switches, 546–547
  STD switch, 547
  usage, 546
POWER program, 35
PREFIX.EXE utility, 320–321
PRINT command, 548–550
  examples, 550
  first time switches, 549
  options and switches, 548–549
  /P switch, 549
  PostScript printers and, 549
  for specifying printing device, 548
  usage, 548
printers
  connecting to shared, 846
  network examples, 849
  remote, accessing, 847–848
processor type, finding, 172
programs
  adding, to menu, 67–68
  BOOTDRV.COM, 365
  CAPSOFF.COM, 360
  CAPSON.COM, 360
  CHOICE.COM, 350–352
  CHOICEAZ.COM, 352
  CURSFAT.COM, 347
    creating, 348

- CURSNORM.COM, 349
- DRIVEAOK.COM, 363–364
- GETFUNCY.COM, 361–362
- KEYSTATE.COM, 358–359
  - ERRORLEVELs set by, 359
- loading, 68
- REBOOT.COM, 361
- starting, 68
- swapping between, 68
- terminating, 68
- WEEKDAY.COM, 356
- WHATDAY.COM, 357
- WHATMON.COM, 357
- WHATYEAR.COM, 357

PROMPT command, 218, 550–552
- character codes, 551
- examples, 551–552
- options and switches, 550
- usage, 550
- using, 220–221

prompts, forcing [Y/N], 264

Protected-Mode, 178
- addressing scheme, 180
- applications, 183

PROTOCOL.INI file, 840
- contents of, 840
- SETUP for modifications of, 841

### — Q —

Q (DEBUG) command, 344

QBASIC, 312
- features, 312

QBASIC command, 46, 552–553
- /EDITOR switch, 553
- examples, 553
- options and switches, 552–553
- /QHELP switch, 553
- usage, 552

QBASIC.EXE file, 19, 312
- changing default wildcard, 355

QEDIT(R) Advanced, 313, 449, 737–751
- registration form, 751

QFIND, 54, 172, 429, 752–759
- registration form, 759

QRAM, 186

Qualitas' 386MAX, 183

Quarterdeck's Expanded Memory Manager, 183

question-mark (?)
- (DEBUG) command, 340
- confirmation prompt, 17
- for forcing [Y/N] prompt, 264
- help, 295
- SWITCHES command and, 264
- *See also* /? switch; ? switch

qz switch, 51

### — R —

R (DEBUG) command, 344

RAM, 161
- cram, 192
- drives, 135–136
- ROM vs., 161–162
- shadow, 184
- *See also* DRAM chip; SRAM chip

RAMDRIVE.SYS, 553–555
- example, 555
- options and switches, 554
- usage, 553

RD command, 555–556
- examples, 556
- functions of, 555
- options and switches, 555
- usage, 555

rebooting, 361

redirection
- to a NUL device, 315–316
- with DOSKEY, 292–293
- example, 315–316
- *See also* piping

redirection characters, 292, 313
- substitutions for, 293

RELDEL utility, 51, 419, 590, 759–760
- registration form, 760
- using, 98

REM command, 307, 556–557
- examples, 557
- options and switches, 556
- usage, 556

RENAME command, 315, 557–558
  examples, 558
  options and switches, 557–558
  renaming directory and, 558
  usage, 557
REPLACE command, 558–560
  ERRORLEVELs, 559–560
  examples, 560
  options and switches, 559
  usage, 558
RESTORE command, 560–562
  date/time variables, format of, 561
  ERRORLEVELs, 562
  examples, 562
  options and switches, 561–562
  usage, 560
RESTORE program, 74
RMDIR command. *See* RD command
ROM, 161
  RAM vs., 161–162
ROM-BIOS, 162
  location of, 192
  shadow RAM and, 184

—S—

S (DEBUG) command, 345
scan codes, 232–233
  understanding, 233
SCAN utility, 25
script files
  ASCII, 349
  BOOTDRV.SCR, 365
  CAPSOFF.SCR, 360
  CAPSON.SCR, 360
  CURSNORM.SCR, 349
  DRIVEOAK.SCR, 363–364
  GETFUNCY.SCR, 361–362
  KEYSTATE.SCR, 358
  REBOOT.SCR, 361
  WEEKDAY.SCR, 356
  WHATDAY.SCR, 357
  WHATMON.SCR, 357
  WHATYEAR.SCR, 357–358
SCSI (Small Computer System Interface), 112, 256
  controller, 112
Search menu, 40
searching
  for files, 54
  for partial directory name, 55–56
  paths, 61
sectors, 114–115
  clusters, 115–116
  dividing disk into, 121
  illustrated, 115
  soft sectoring and, 121
  *See also* tracks
SeekEasy utility, 469, 761–769
  registration form, 769
SET command, 208, 318, 563–564
  examples, 564
  options and switches, 563
  unallowed varname characters, 563
  usage, 563
SET files, in backup, 82
SETUP, 72
  backup file selection from, 75
  /E switch, 812
  /M switch, 812
  for PROTOCOL.INI modifications, 841
  /Q switch, 813
  supported network cards display, 836
  Workgroup Connection settings display, 838
SETUP command, 36, 564–565
  example, 565
  options and switches, 564–565
  /Q switch, 564, 565
  usage, 564
SETVER command, 566–567
  /D switch, 566
  examples, 567
  options and switches, 566–567
  usage, 566
SETVER.EXE command, 567–568
  example, 568
  options and switches, 568
  usage, 567

SHARE command, 568–569
  example, 569
  loading, 569
  options and switches, 568–569
  usage, 568
  uses, 569
Shareware programs, 6–7, 605–606
  ASP and, 607
    members, 607
    ombudsman program, 608
    support, 608
  DIR alternatives, 57–59
  license agreement, 608–609
  registration of, 7
    benefits, 606–607
  where to get, 607
  *See also* Bob's Better-Than-DOS Shareware Collection
SHELL command, 208, 569–570
  COMMAND.COM and, 570
  example, 570
  MEMMAKER and, 570
  options and switches, 569–570
  usage, 569
SHIFT command, 307, 570–571
  in batch files, 571
  example, 571
  options and switches, 570
  usage, 570
sidebars, 8
signature characters, 218
SIMMs (single in-line memory modules), 165
  illustrated, 166
SIZER command, 571
SLT files, in backup, 82
SMARTDRV.EXE, 572–575
  buffersize variable, 572
  as device driver, 574
  DoubleSpace volumes and, 574
  elementsize variable, 572
  examples, 574
  extended use, 574–575
  for identifying double buffering, 574
  initcachesize/wincachesize default values, 573
  installing in low memory, 573
  options and switches, 572–573
  usage, 572
  write caching, 575
SMARTDRV program, 33, 129
  caching compressed drives with, 15
  cluster problems with, 134
  default cache sizes, 131
  with DoubleSpace
    installed, 133
    using, 134
  FASTOPEN and, 136
  memory allocation, 129
  resolving communications problems with, 133
  using, 129–133
    with CD-ROMs, 134
    with DoubleSpace drives, 134
  *See also* disk caches
SMARTMON utility, 34, 46
soft sectoring, 121
SORT command, 575–576
  > and < characters, 576
  examples, 576
  options and switches, 575–576
  usage, 575
SpinRite program, 128
SpinTest program, 127, 769–772
SpinTime program, 127, 769–772
SRAM chip, 164–165
  *See also* DRAM chip; RAM
STACKER
  clean boot and, 263
  DoubleSpace drives and, 147
STACKEY utility, 31, 41–42, 773–785
  ECHO comparison, 314–315
  registration form, 785
STACKS command, 576–577
  examples, 577
  options and switches, 577
  stack error messages and, 577
  usage, 577
start-up menus, 265
  changing colors for, 268–269
  changing headings for, 269

  commands for, 266
  CONFIG.SYS file for, 266–267
  defining, 266–269
  entries per menu, 267
  setting defaults for, 268
  *See also* multimenu configurations
Stealth, 184
strings, assigning to keys, 237–239
subdirectories, listing, 53
SUBMENU command, 15, 578–579
  blockname variable, 578
  example, 578–579
  options and switches, 578
  syntax, 272
  usage, 578
SUBST command, 579–580
  examples, 580
  options and switches, 579
  usage, 579
  Windows and, 580
swap files, 149
  problems with, 149–150
SWITCH command, 580–581
  example, 581
  /K switch, 581
  options and switches, 580–581
  usage, 580
  /W switch, 580, 581
switches, 9
  ANSI.SYS, 373–374
  APPEND command, 375–376
  ATTRIB command, 378–379
  BREAK command, 381
  BUFFERS command, 383
  CALL command, 385
  CD command, 387
  CHCP command, 389
  CHKDSK command, 390
  CHOICE command, 393–394
  CLS command, 396
  COMMAND command, 398–399
  command-line, 291–292
  COPY command, 401
  COUNTRY command, 404
  CTTY command, 405
  DATE command, 406
  DBLSPACE command, 408–410
  DBLSPACE command line, 153
  DBLSPACE.SYS, 412
  DEBUG command, 413–414
  DEFRAG command, 414–416
  DEL command, 417–418
  DELOLDOS command, 419
  DELTREE command, 420–421
  DEVICE command, 422
  for device drivers, 17
  DEVICEHIGH command, 423–424
  DIR command, 56–57, 426–428
  DISKCOMP command, 430
  DISKCOPY command, 431–432
  DISPLAY.SYS, 433–434
  DOS command, 435
  DOSKEY command, 437–438
  DOSSHELL command, 440
  DRIVER.SYS, 441–442
  DRIVPARM command, 444–445
  ECHO command, 446
  EDIT command, 448–449
  EGA.SYS, 450
  EMM386 command, 451–455
  EXIT command, 457
  FASTHELP command, 17, 459
  FASTOPEN command, 460
  FC command, 462–463
  FCBS command, 465
  FDISK command, 465–466
  FILES command, 467
  FIND command, 468
  FOR command, 470–471
  FORMAT command, 472–474
  GOTO command, 476
  GRAPHICS command, 477–479
  HELP command, 480
  HIMEM.SYS, 482–484
  IF command, 485
  INCLUDE command, 487
  INSTALL command, 489
  INTERLNK command, 490
  INTERLNK.EXE, 492–493
  INTERSVR command, 494–495

- KEYB command, 497
- LABEL command, 498–499
- LASTDRIVE command, 500
- LOADFIX command, 501
- LOADHIGH command, 502
- MAIL command, 856–857
- MD command, 504
- MEM command, 505–506
- MEMMAKER command, 507–508
- MENUCOLOR command, 510
- MENUDEFAULT command, 511
- MENUITEM command, 512–513
- MICRO command, 853, 858–859
- MIRROR command, 514–515
- MODE COMn command, 517–518
- MODE CON command, 519
- MODE device CP command, 520
- MODE device /STATUS command, 521
- MODE display command, 522
- MODE LPTn command, 523
- MORE command, 525
- MOVE command, 526
- MSAV command, 528–529
- MSBACKUP command, 530–531
- MSCDEX command, 533
- MSD command, 535
- MWUNDEL command, 538
- NET CONFIG command, 861
- NET LOGOFF command, 862
- NET LOGON command, 863
- NET PASSWORD command, 865
- NET PRINT command, 866–867
- NET START command, 868–689
- NET STOP command, 870
- NET TIME command, 871–872
- NET USE command, 872–874
- NET VER command, 876
- NET VIEW command, 876–877
- NLSFUNC command, 540
- NUMLOCK command, 541
- PATH command, 542
- PAUSE command, 543
- POWER command, 545
- POWER.EXE, 546–547
- PRINT command, 548–549
- PROMPT command, 550
- QBASIC command, 552–553
- RAMDRIVE.SYS, 554
- RD command, 555
- REM command, 556
- RENAME command, 557–558
- REPLACE command, 559
- RESTORE command, 561–562
- SET command, 563
- SETUP command, 564–565
- SETVER command, 566–567
- SETVER.EXE, 568
- SHARE command, 568–569
- SHELL command, 569–570
- SHIFT command, 570
- SMARTDRV.EXE, 572–573
- SORT command, 575–576
- STACKS command, 577
- SUBMENU command, 578
- SUBST command, 579
- SWITCHES command, 580–581
- SYS command, 582
- TIME command, 583
- TREE command, 585
- TYPE command, 586
- UNDELETE command, 587–589
- UNFORMAT command, 592
- UNINSTAL command, 591
- VER command, 594
- VERIFY command, 594
- VOL command, 595
- VSAFE command, 596–597
- XCOPY command, 599–600
- *See also* DOS switches

SWITCHES command, 16, 264
- ? character and, 264
- controlling start keys with, 264–265
- switches, 265

SYS command, 582–583
- examples, 582
- options and switches, 582
- usage, 582

## —T—

T (DEBUG) command, 345
task switching, 67–68
  performing, operations, 67–68
TIME command, 583–584
  CONFIG.SYS country setting and, 583
  examples, 584
  feeding Enter to, 324
  FIND command and, 325, 326
  no battery-supported clock and, 584
  options and switches, 583
  for pausing batch file until time is reached, 325–326
  usage, 583
tracks, 114
  illustrated, 115
  *See also* sectors
TREE command, 584–586
  examples, 585
  options and switches, 585
  usage, 584
  | MORE pipe, 585
TSRs, 88–89
  checking for, 330–331
  Windows required, 333
TURBOBAT utility, 46, 336, 386, 396, 477, 487, 571
  batch file tampering prevention, 544
  *See also* BATUTIL utility
TURBOTXT utility, 679–680
TYPE command, 218, 586–587
  example, 586
  limitations, 586
  options and switches, 586
  usage, 586
  using, 220

## —U—

U (DEBUG) command, 345
UART chip, 133
  history of, 133
UMB_DRV_SYS, 424, 436, 456, 503, 785–799
  for advanced memory support, 509
  registration form, 800
UMBs (upper memory blocks), 184
  CONFIG.SYS file for using, 191
  controlling load order and location, 193–194
  cram, 192
  creating, 190
  identifying, space, 192–193
  INTERLNK.EXE and, 492
  loading programs into, 191
  memory allocation, 194–195
  regions, 193–194
  using, 190–192
  *See also* upper memory
UNDELETE command, 31, 32–33, 587–590
  Delete Sentry and, 589
  Delete Tracker and, 589
  deleted directories and, 589
  DOS and, 589
  examples, 590
  options and switches, 587–589
  switches for undeleting files, 588
  usage, 587
UNDELETE.INI file, 90–91, 588
  example of, 90
  sections of, 90
UNDELETE utility, 31, 32–33, 72
  delete sentry method, 89
  delete tracker method, 89
  deleted file list with, 94
  file protection methods, 33
  loading, 88
  TSR, 88–89
undeleting, 95
  files, 93–94
UNFORMAT command, 31, 95–97, 592–593
  examples, 593
  limitations, 593
  options and switches, 592
  usage, 592
UNFORMAT utility, 72, 95–97
  limitations, 97
  MIRROR files and, 96–97
  recovering directories with, 96
  recovering files with, 97

unformatting, 95–97
  process of, 96
UNINSTAL command, 147, 591–592
  ? switch and, 591
  limitations, 591
  options and switches, 591
  usage, 591
UNINSTALL program, 419, 420
  DBLSPACE and, 420
uninstalling DOS 6, 813
upper memory, 131–132
  address space, 172
  B area of, 201
  defined, 172
  loading DBLSPACE.BIN into, 195
  loading DOS into, 189–190
  monochrome region of, 201
  optimizing, for use with Windows, 200–201
  showing loaded locations in, 204
  utilization of, 193
  *See also* UMBs (upper memory blocks)
utilities. *See* disk utilities; windows-hosted utilities

—V—

VCPI (Virtual Control Program Interface), 184
VER command, 330, 593–594
  example, 594
  options and switches, 594
  usage, 593
VERIFY command, 594–595
  examples, 595
  options and switches, 594
  usage, 594
VESA (Video Electronics Standards Association), 256
  VL bus standard, 256
virtual memory, 184–185
virus(es)
  classification types of, 99–100
  cleaning, 103
  defined, 98
  functioning of, 99–100
  guarding against, 102–103
  identifying, 101–103
  measures, 98–100
  on-line info, 104
  phenomenon, 98–99
  potential damage caused by, 100
  protection, 24–25, 72
  removing, 103–104
  scanning for, 101
  signatures, 24
    updating, 25
VIRUSCAN utility, 800–805
  registration form, 805
VOL command, 595–596
  example, 596
  options and switches, 595
  usage, 595
VSAFE command, 596–598
  /AX switch, 597
  /CX switch, 597
  examples, 598
  option codes, 597
  options and switches, 596–597
  uninstalling, 597
  usage, 596
VSAFE program, 24, 72, 102–103
  function of, 25, 98, 598
  with Windows, 103

—W—

W (DEBUG) command, 345
wait state, 167
  zero, 167
wildcards
  changing default in QBASIC.EXE, 355
  checking for, 329–330
  entering, 372
Windows 3.1, Microsoft, 332–334
  adding WINPMT variable, 333
  batch files with, 332–334
  checking for, 334
  commands not good with, 334
  DOS session settings and, 333
  /K switch and, 334
  lowercase variable, 334

    suppressing default message, 332
    windir variable, 334
    WINSTART.BAT, 333
    *See also* Windows, Microsoft
Windows for Workgroups, 836, 837
Windows, Microsoft
    Anti-Virus, 72, 73
    APPEND command and, 378
    Backup option, 72, 73
    Enhanced mode performance improvement, 135
    Fastdisk, using with, 135
    installing
        applications, 72
        components of DOS 6, 812
    optimizing upper memory for, 200–201
    Undelete option, 72–73
    utilities, 33–34
    VSAFE with, 103
    *See also* Windows 3.1, Microsoft
windows-hosted utilities, 33–34
Workgroup Connection, 835
    command reference, 855–877
    settings, 838
    software, 838
        installing, 837
    using, 835–836
    *See also* Workgroup Connection commands
Workgroup Connection commands, 855–877
    MAIL, 850, 856–858
    MICRO, 853–855, 858–860
    NET, 838–839, 860–861
    NET CONFIG, 861–862
    NET LOGOFF, 862–863
    NET LOGON, 842, 863–864
    NET PASSWORD, 865–866
    NET PRINT, 849, 866–868
    NET START, 841–842, 868–870
    NET STOP, 870–871
    NET TIME, 871–872
    NET USE, 847–848, 872–875
    NET VER, 876
    NET VIEW, 846–847, 876–877
write buffering, 33
    *See also* buffering

—X—

XA (DEBUG) command, 346
XCOPY command, 17, 47–48, 598–601
    /E switch, 600
    ERRORLEVELs, 600
    examples, 600–601
    options and switches, 599–600
    reasons to use, 47
    /S switch, 599
    switches supported by, 47
    usage, 599
    *See also* COPY command; PCOPY utility
XD (DEBUG) command, 346
XM (DEBUG) command, 346
XMS (extended memory specification), 179
XS (DEBUG) command, 346

—Z—

zone bit recording, 115
    explained, 116

# Notes

  improving reliable operation, 100
  mode switches, 102
  WINSTART.BAT, 102
  See also Windows, Microsoft
Windows for Workgroups, 828-829
Windows, Microsoft
  AntiVirus, 55-57
  APPEND command and, 799
  Backup options, 38, 44
  Enhanced mode (protected mode)
      implementation, 109
  Expanded memory and, 108
  installing
    applications, 78
    independently of DOS, 843
  optimizing upper memory for, 192-193
  Undelete options, 70-73
  utilities, 39-44
  VSAFE with, 100
  See also Windows 3.1, Microsoft
windows-hosted utilities, 39-44
Workgroup Connection, 828
  command reference, 858-877
  settings, 825
  software, 828
    installing, 827
  using, 826-828
  See also Workgroup Connection
      commands
Workgroup Connection commands, 858-877
  NET, 859, 858-859
  NET HELP, 829-830, 860-861
  NET INIT, 861-862
  NET LOGOFF, 862-863
  NET LOGON, 830, 863-864
  NET PASSWORD, 865-866
  NET PRINT, 830, 866-868
  NET START, 831-832, 868-870
  NET STOP, 870-871
  NET TIME, 871-873
  NET USE, 831-832, 873-875
  NET VER, 876
  NET VIEW, 831-832, 875-876
write-behind, 30
  See also caching

—X—

XALLOC(EMS) command, 795
XCOPY command, 17, 37, 45, 596-600
  /E switch, 600
  ERRORLEVELS, 600
  examples, 599-601
  options and switches, 596-598
  reasons to use, 37
  /S switch, 599
  switches supported by, 47
  usage, 598
  See also COPY command; PCOPY utility
XMA(EMS) command, 796
XM(DEM) command, 796
XMS (extended memory specification), 178
XS(STATUS) command, 796

—Z—

zone bit recording, 115
  explained, 116

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

# Notes

To learn even more about DOS, subscribe to

# *DOS* Resource Guide

The only magazine devoted *exclusively* to solving the difficult DOS problems you face every day.

If you have questions about: optimizing CONFIG.SYS, writing a batch file, or learning to program in QBasic, you need DOS Resource Guide, the magazine with answers to all of your toughest DOS problems.

Each issue of DOS Resource Guide is filled with undocumented shortcuts and Super Tips on using DOS, along with batch files and QBasic programs you can type in and use.

Subscribe *now* and save up to 25% off the newstand cover price!

Use the coupon below, or call toll-free: **800-238-6580**

☐ YES! Send me One Year (6 issues) of DOS Resource Guide for $23.70*

☐ YES! Save me EVEN MORE! I'll take Two Years (12 issues) of DOS Resource Guide for $44.00*. That saves me 25% off the cover price!

Name: ______________________________

Address: ______________________________

City: ____________________ State: ________ Zip: __________

(Check, Credit Card, or Money Order — Payment must accompany your order.)

☐ VISA ☐ MasterCard ☐ AMEX ☐ Discover

Credit Card # ____________________ Exp. Date: __________

Signature: ______________________________

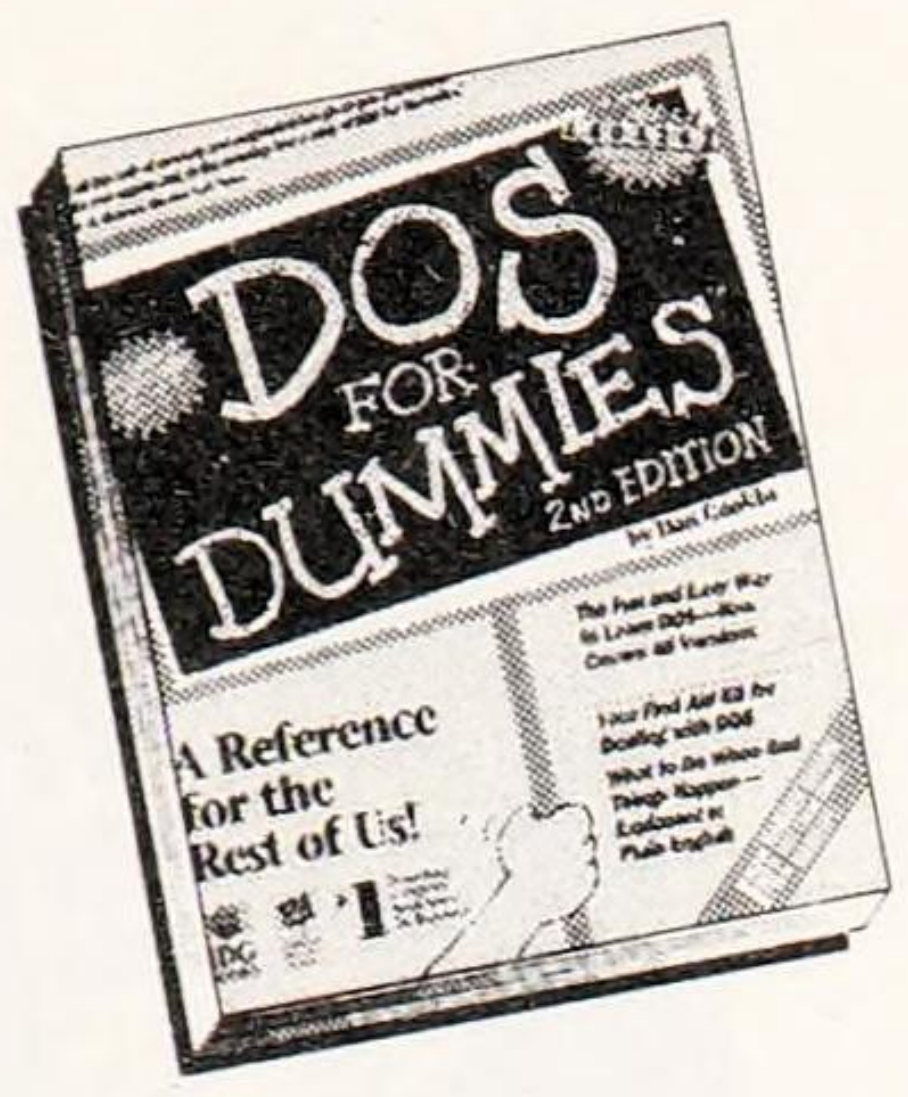
Mail to: DOS Resource Guide, 80 Elm Street, Peterborough NH 03458

* Maryland residents add 5% sales tax. Canada and Mexico: $29.70 (one year), $55.00 (two years), includes GST. All Other Foreign Orders: Surface mail $41.70 (one year), $78.00 (two years). Air mail $53.70 (one year), $99.00 (two years). All foreign orders must be paid in U.S. funds drawn on a U.S. bank. DFD2-93

...FOR DUMMIES™
COMPUTER BOOK SERIES FROM IDG

# ...For Dummies Computer Book Series™

The #1 Computer Books Series For Beginners — over 2,000,000 books in print!. These books end the frustrations of beginning computer users who know they aren't dumb, but find that PC hardware, software, and computer jargon make them feel helpless. Lighthearted, down-to-earth, and packed with valuable information ...*For Dummies* Books are the perfect survival guides for anyone forced to use a computer.

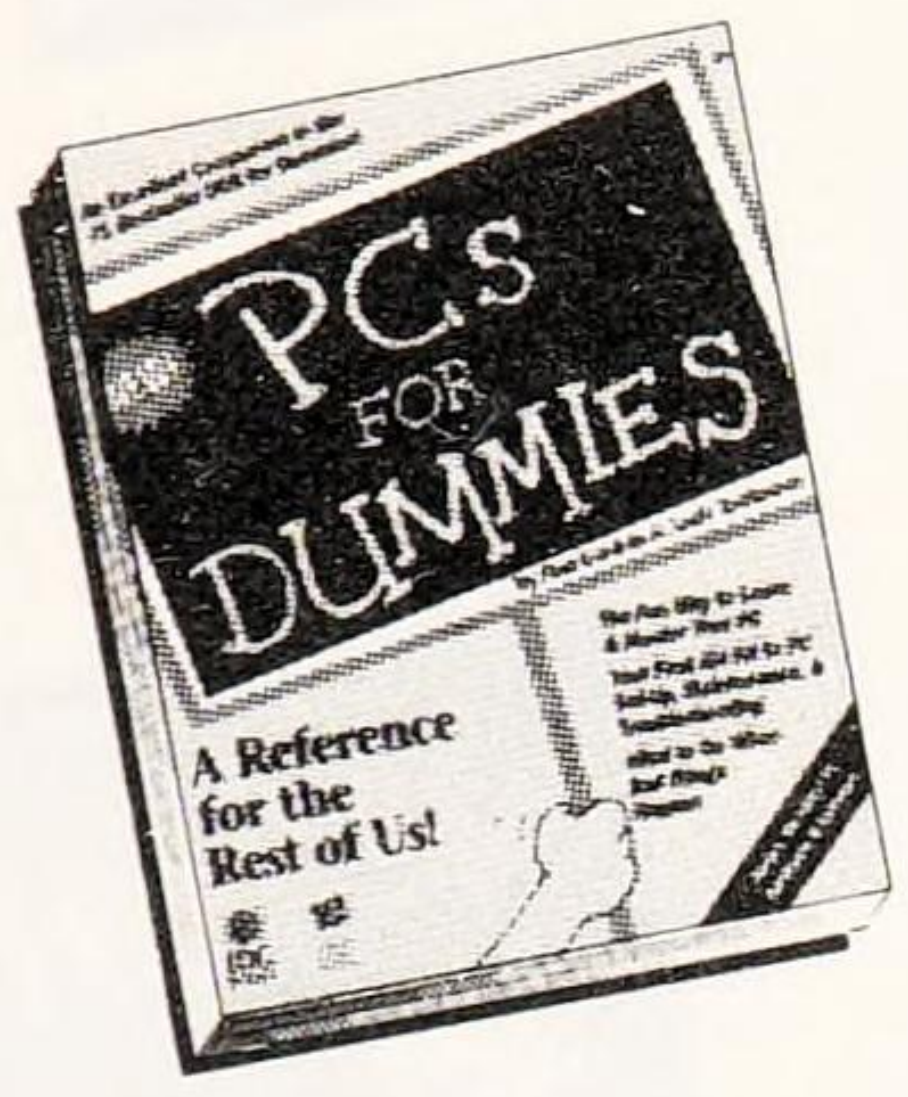


## DOS For Dummies,™ 2nd Edition

*by Dan Gookin*

Learn DOS the fun and easy way with Dan Gookin's humorous reference for the rest of us – covers DOS ver. 2-6.

**$16.95 USA/$21.95 Canada/£15.45 UK & EIRE, ISBN: 1-878058-75-4**

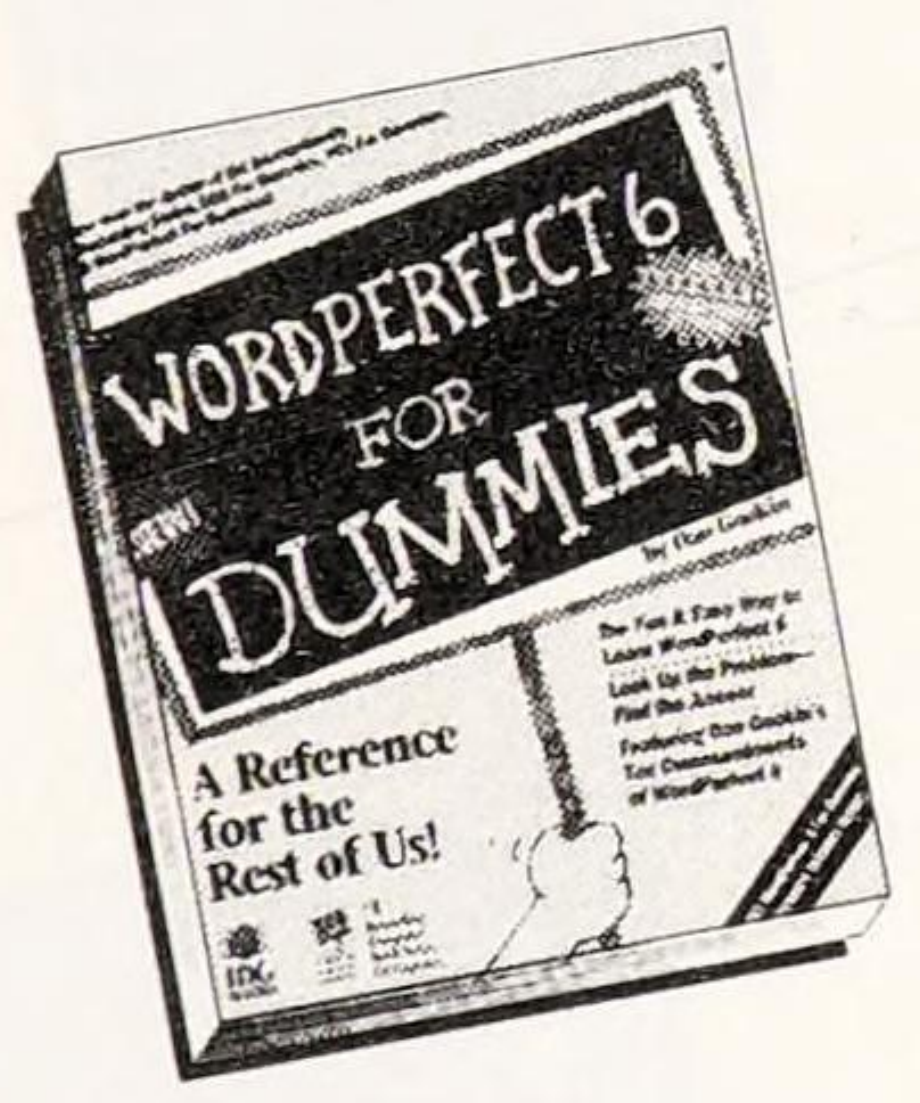


## PCs For Dummies™

*by Dan Gookin and Andy Rathbone*

The non-nerd's guide to PC configuration, upgrading, and repair – for the computer phobic!

**$16.95 USA/$21.95 Canada/£15.45 UK & EIRE, ISBN: 1-878058-51-7**

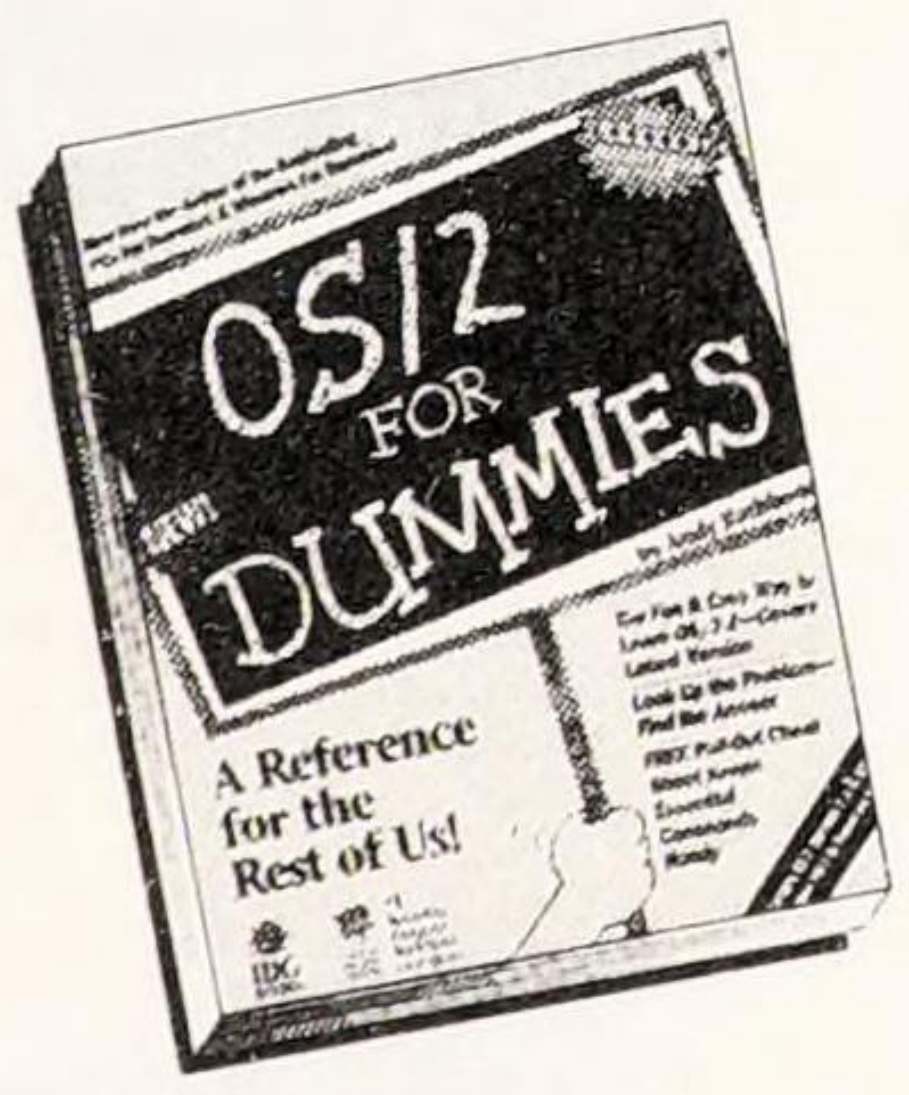


## WordPerfect 6 For Dummies™

*by Dan Gookin*

The sequel to the international bestseller *Word Perfect For Dummies,* now completely revised for WordPerfect 6.0.

**$16.95 USA/$21.95 Canada/£15.45 UK & EIRE, ISBN: 1-878058-77-0 — Available Aug. 1993**

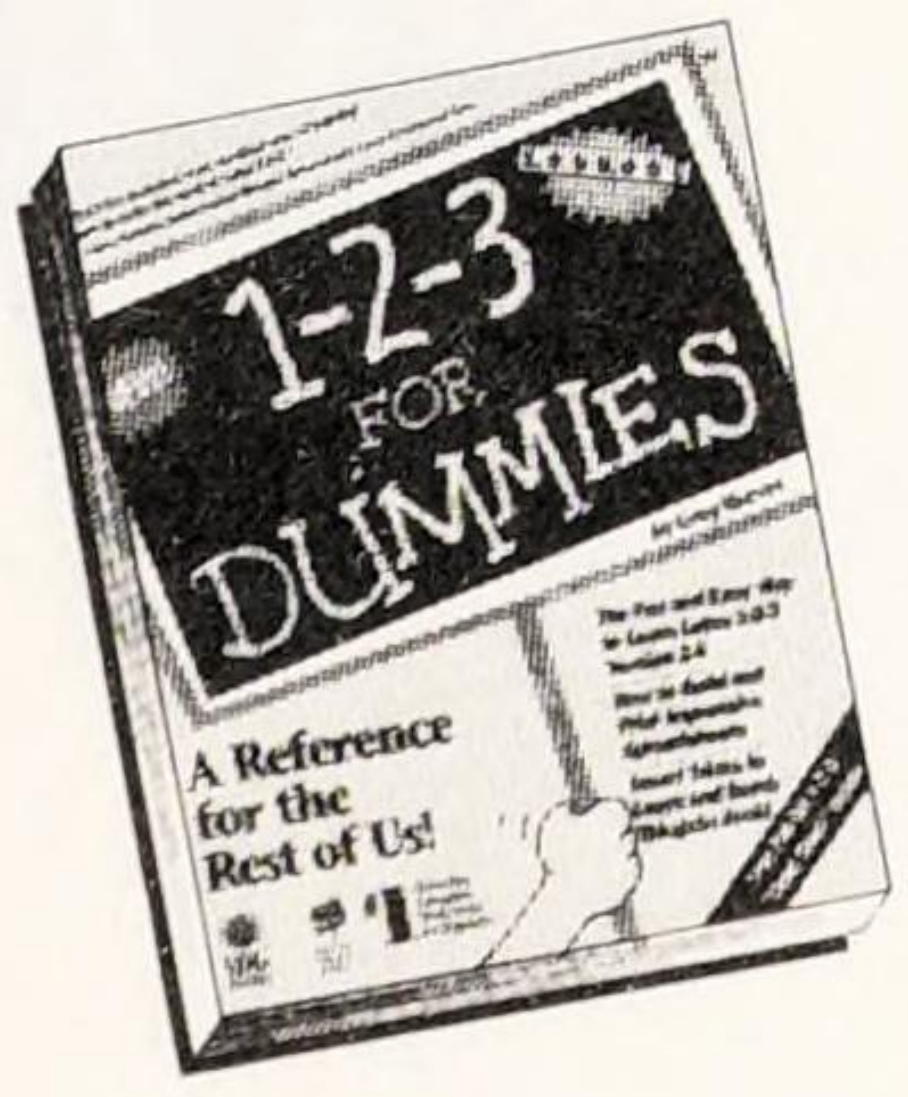


## OS/2 For Dummies™

*by Andy Rathbone*

Bestselling *PCs For Dummies* author Andy Rathbone offers a fun and easy survival guide to beginning and intermediate users who need to co-exist with OS/2.

**$19.95 USA/$26.95 Canada/£18.45 UK & EIRE, ISBN: 1-878058-76-2**

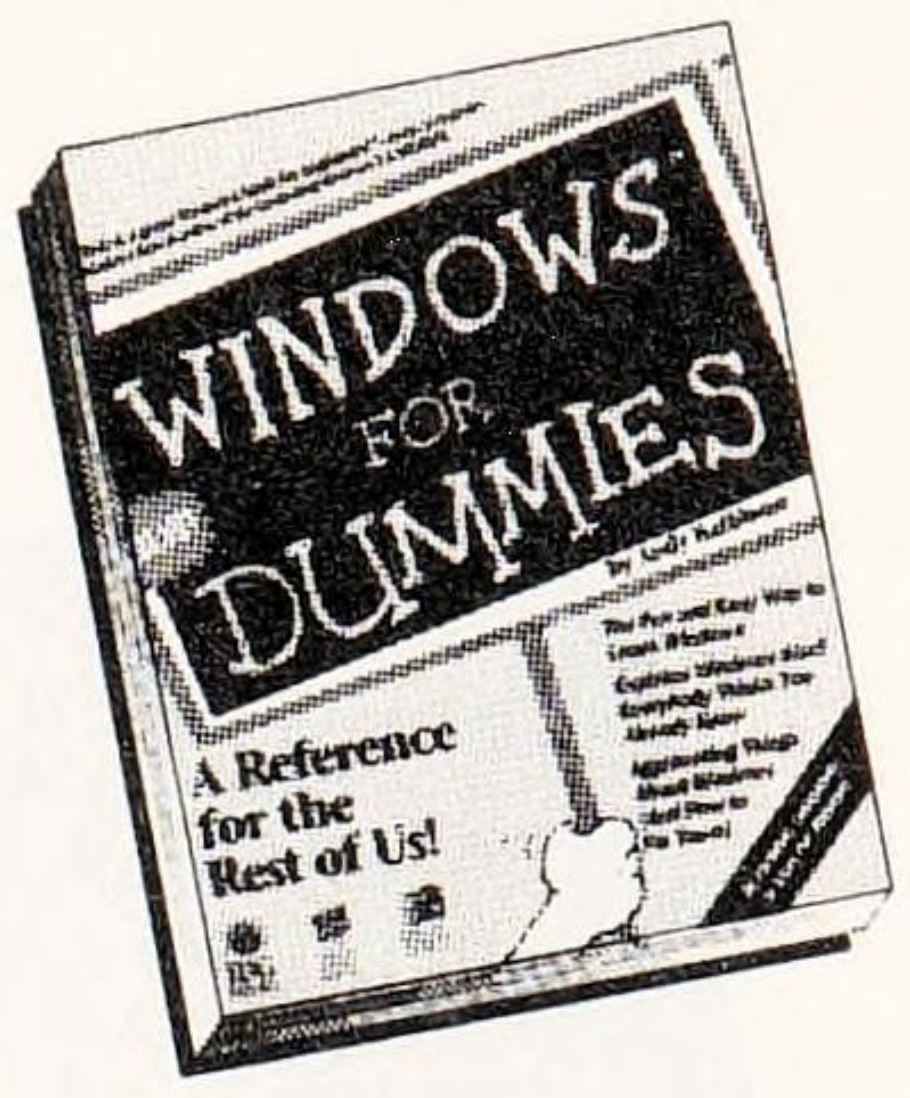


## 1-2-3 For Dummies™

*by Greg Harvey*

Simplifies Lotus 1-2-3, the perfect introduction for the spreadsheet averse.

**$16.95 USA/$21.95 Canada/£15.45 UK & EIRE, ISBN: 1-878058-60-6**

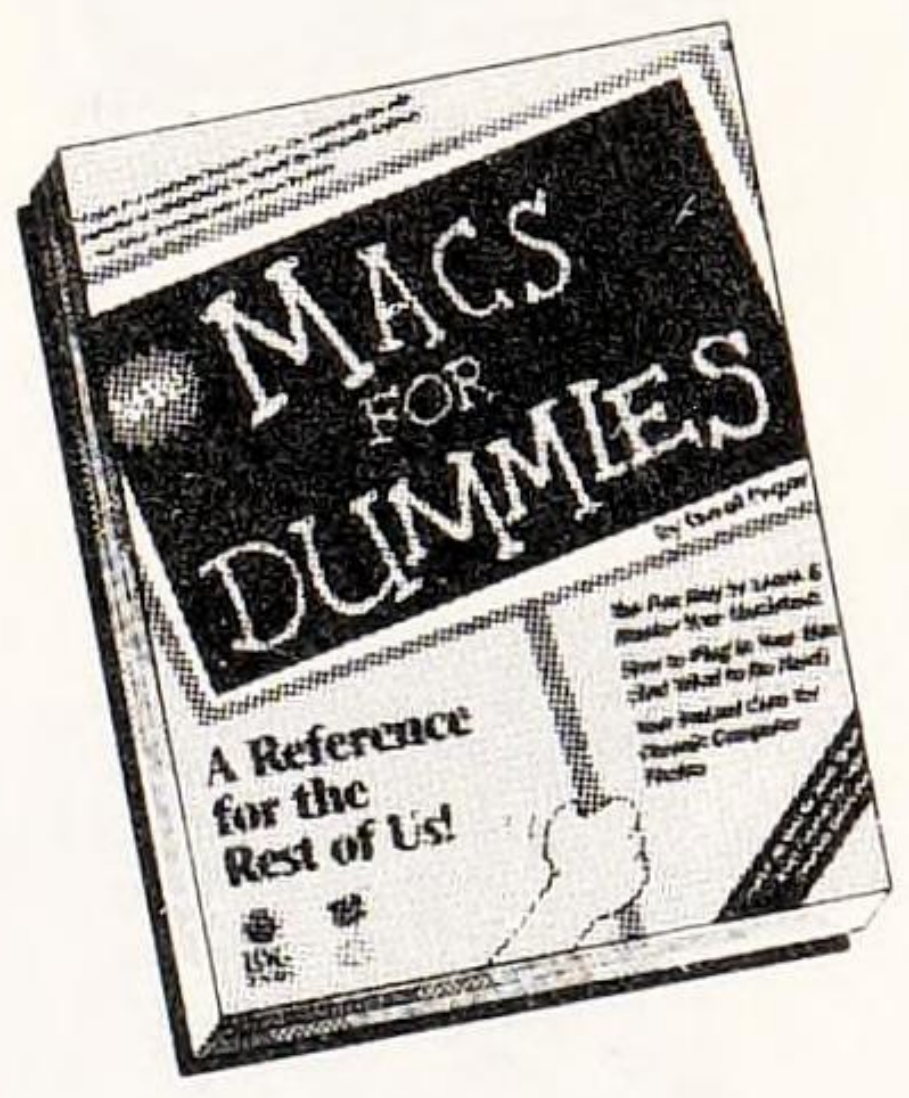


## Windows For Dummies™

*by Andy Rathbone, coauthor of the bestselling PCs For Dummies*

Teaches new Windows users the essentials of Windows with humor and style!

**$16.95 USA/$21.95 Canada/£15.45 UK & EIRE, ISBN: 1-878058-61-4**

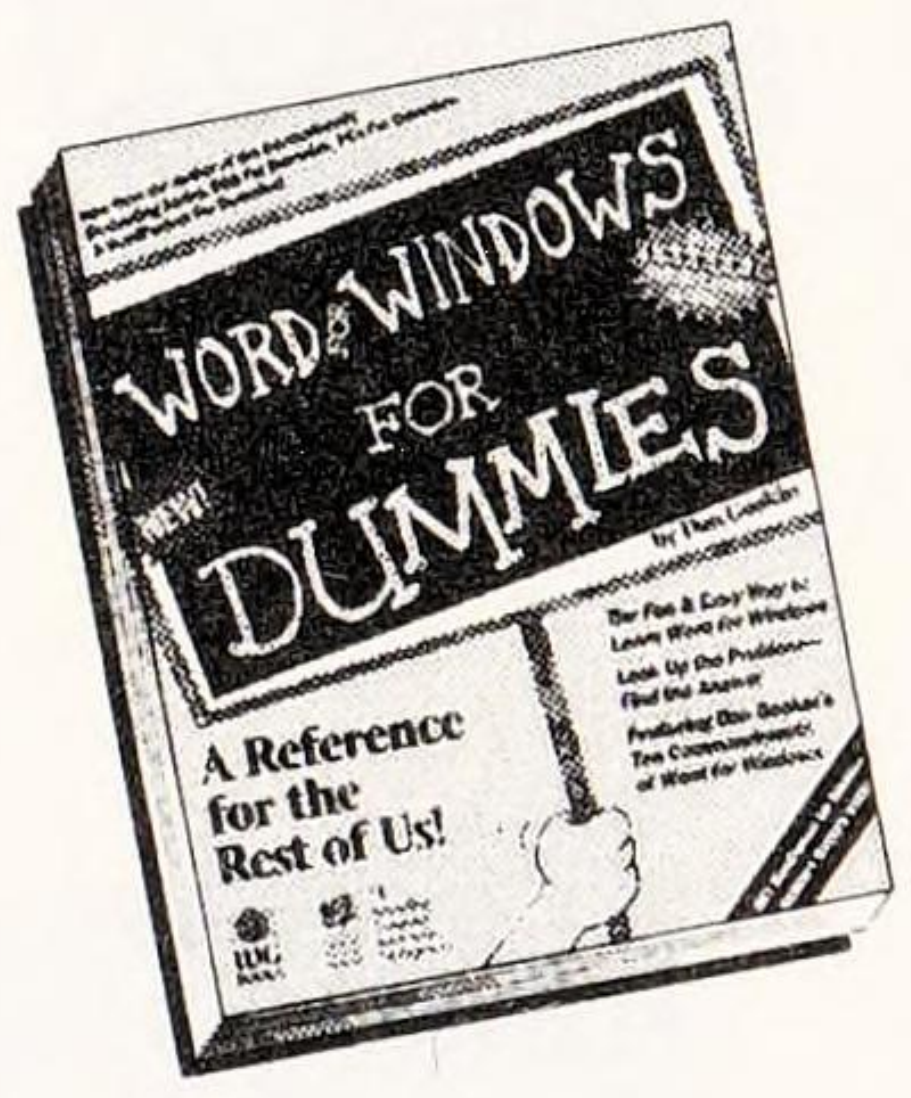


## Macs For Dummies™

*by David Pogue*

Finally, a friendly guide to plugging in and learning how to use your Mac!

**$16.95 USA/$21.95 Canada/£15.45 UK & EIRE, ISBN: 1-878058-53-3**

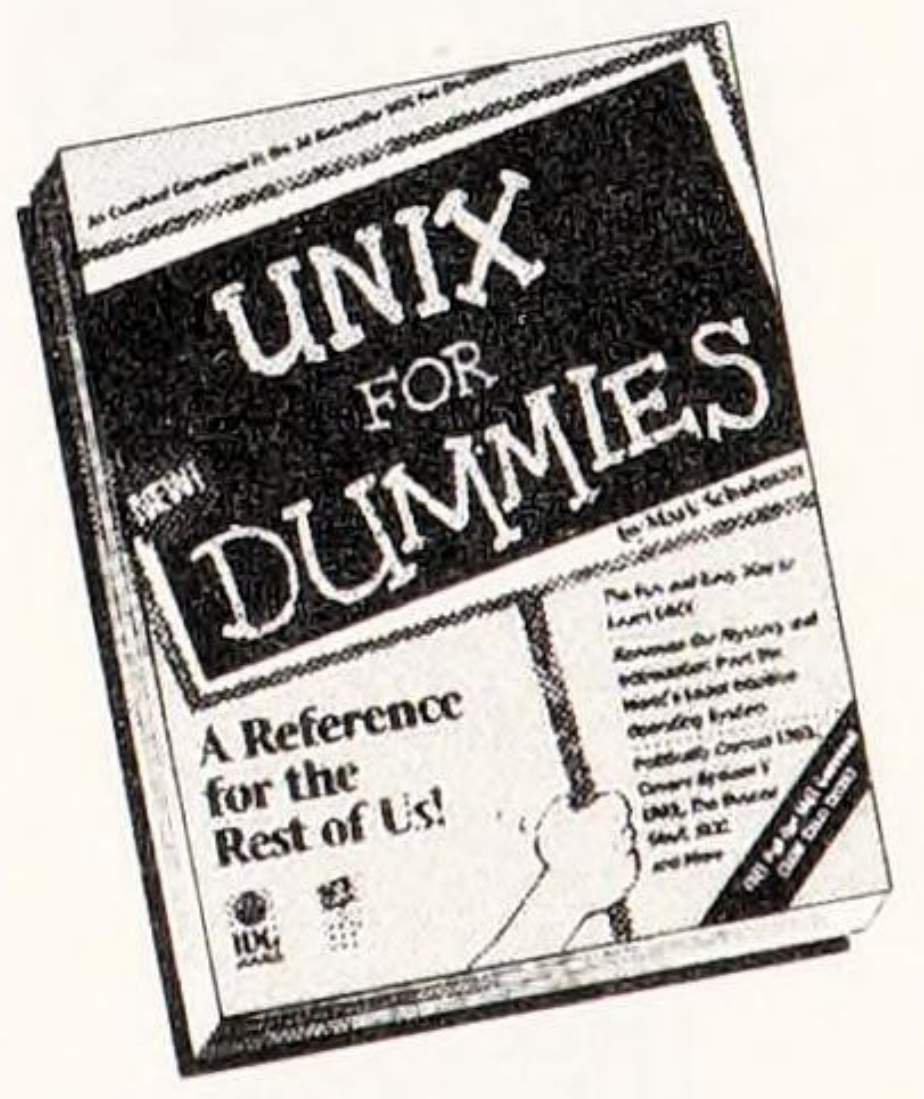


## Word for Windows For Dummies™

*by Dan Gookin*

Bestselling author Dan Gookin brings his lighthearted ...*For Dummies* approach to Word for Windows—today's fastest growing word processing program.

**$16.95 USA/$21.95 Canada/£15.45 UK & EIRE, ISBN: 1-878058-86-X — Available June 1993**

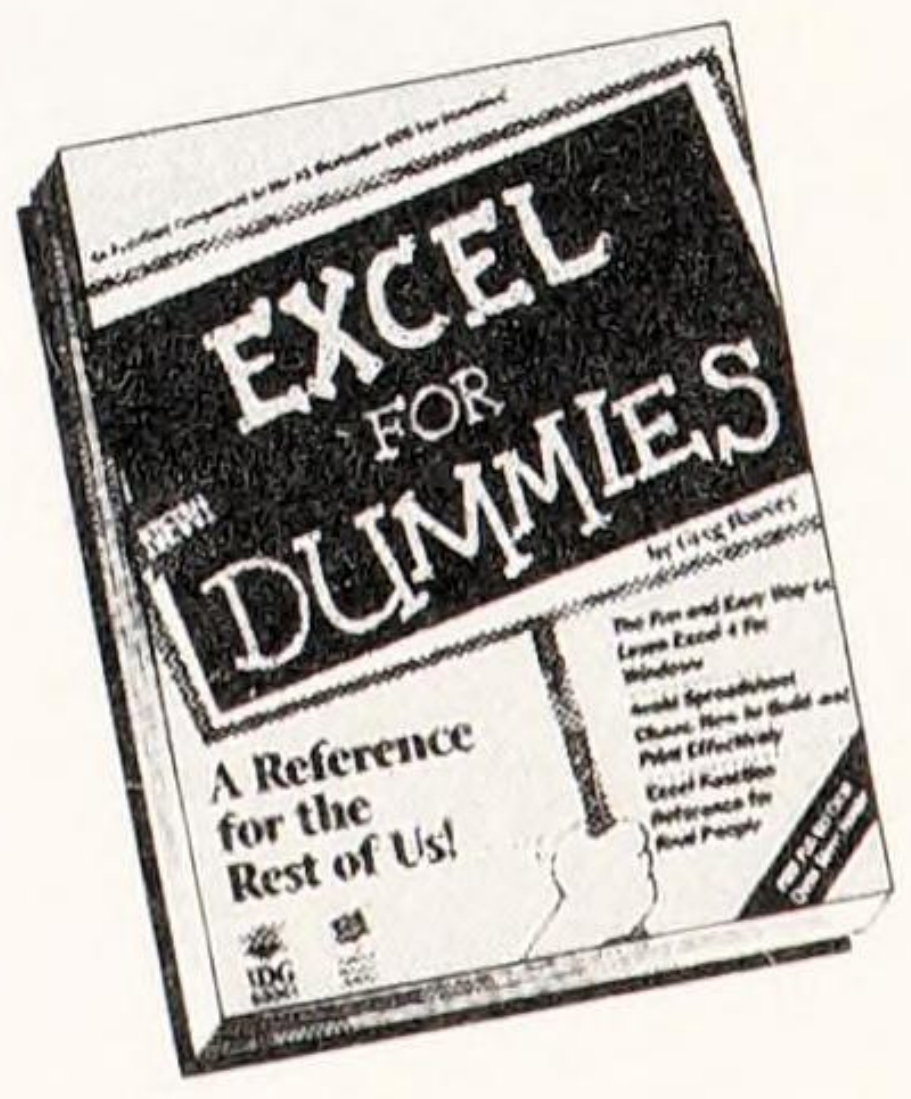


## UNIX For Dummies™

*by John Levine*

The fun and friendly guide that takes the mystery out of UNIX.

**$19.95 USA/$26.95 Canada/£18.45 UK & EIRE, ISBN: 1-878058-58-4**

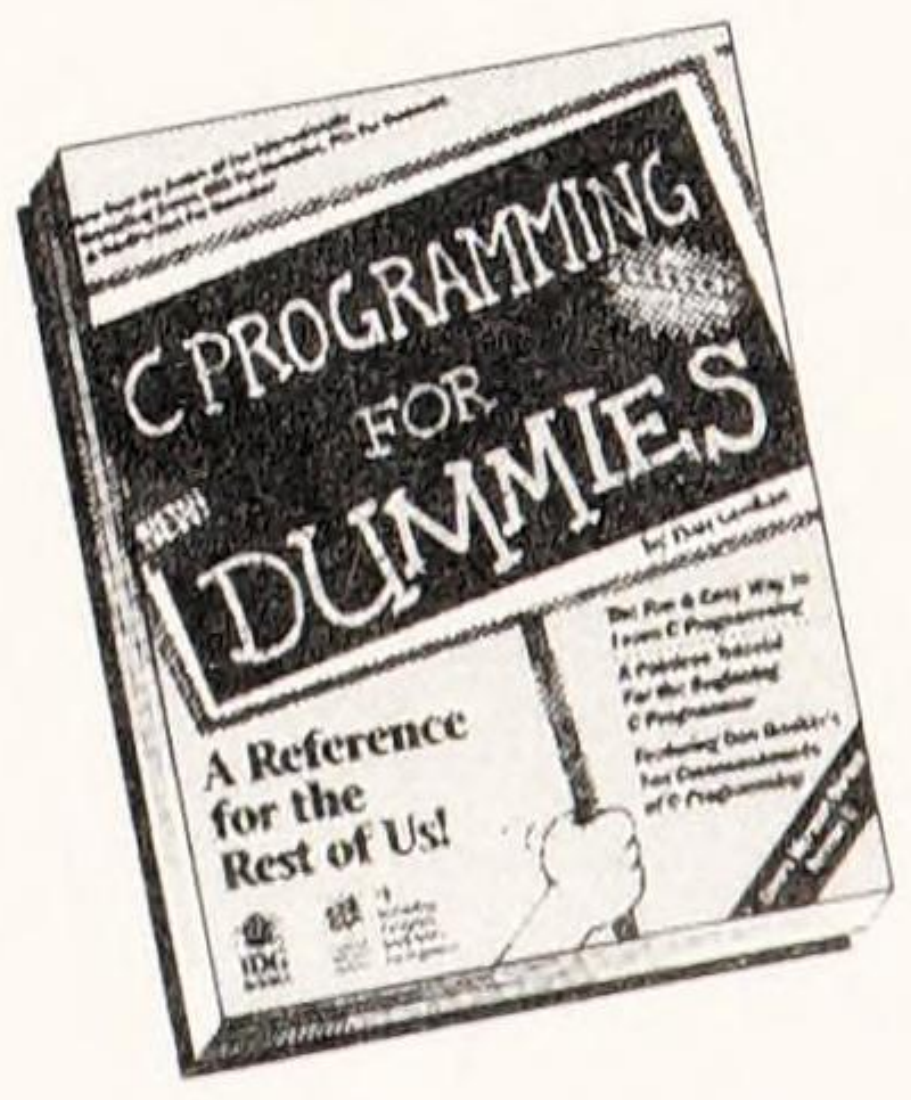


## Excel For Dummies™

*by Greg Harvey*

The best beginner's guide to Excel 4 for Windows —today's fastest growing spreadsheet program.

**$16.95 USA/$21.95 Canada/£15.45 UK & EIRE, ISBN: 1-878058-63-0**



## C Programming For Dummies™

*by Dan Gookin, author of the #1 Bestseller DOS For Dummies*

A painless primer – unique, entertaining style to teach the essentials of C Programming. Guaranteed to appeal to even the most cautious C novice. Never stuffy, this hands-on tutorial is loaded with easy and entertaining examples.

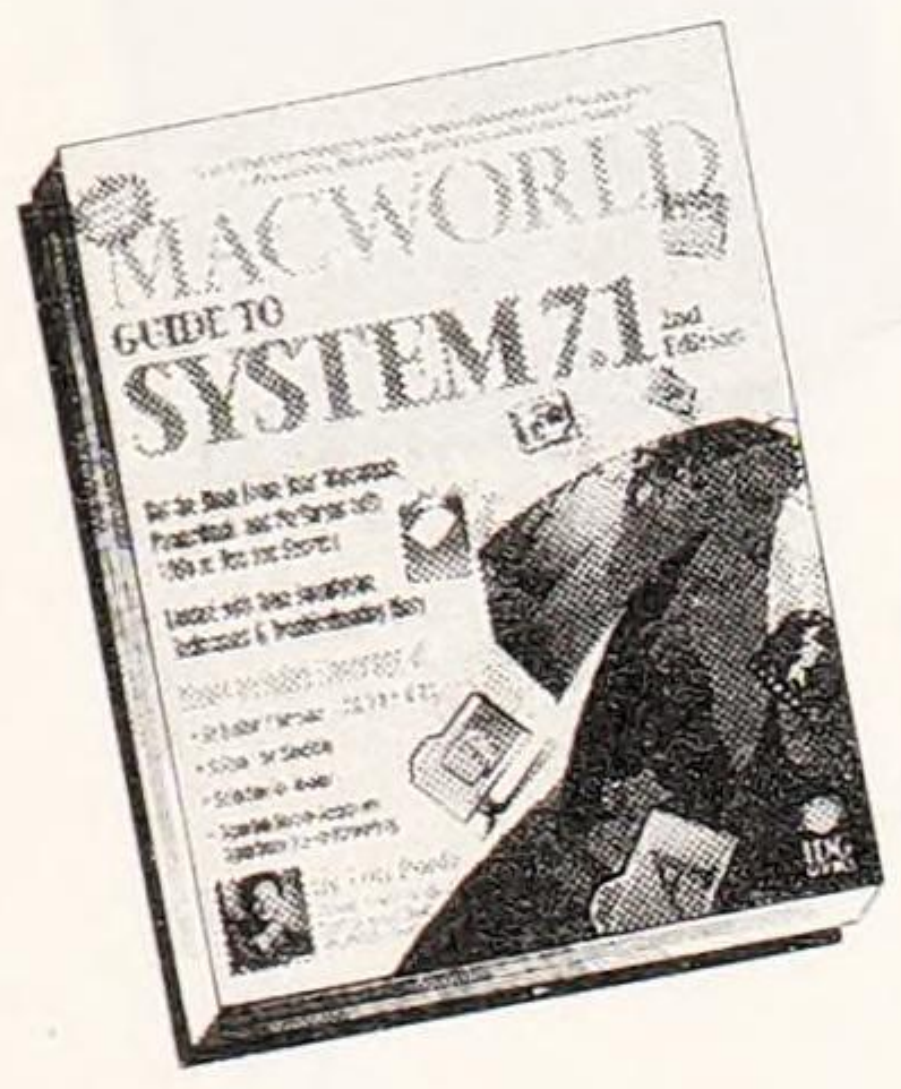
**$19.95 USA/$26.95 Canada/£18.45 UK & EIRE, ISBN: 1-878058-78-9 — Available June 1993**

*For More Information Call 1-800-762-2974*

MACWORLD
MW
AUTHORIZED
EDITION

# Macworld Authorized Editions

Designed specifically for the Macintosh user, Macworld Books are written by leading *Macworld* magazine columnists, technology champions, and Mac gurus who provide expert advice and insightful tips and techniques not found anywhere else. Macworld Books are the only Macintosh books authorized by *Macworld*, the world's leading Macintosh magazine.

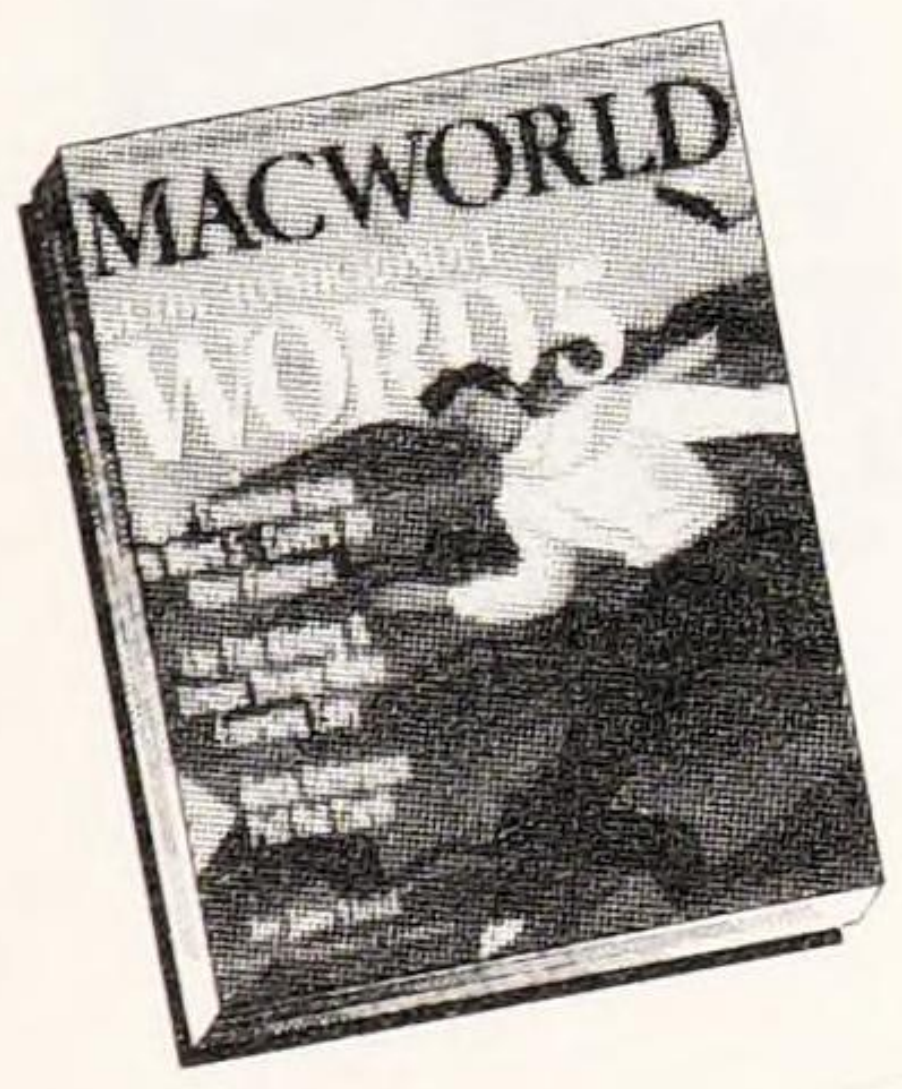


### Macworld Guide To Microsoft System 7.1, 2nd Edition

*by Lon Poole, Macworld magazine's "Quick Tips" columnist*

The most recommended guide to System 7, updated and expanded!

**$24.95 USA/$33.95 Canada/£22.92 UK & EIRE,**
**ISBN: 1-878058-65-7**

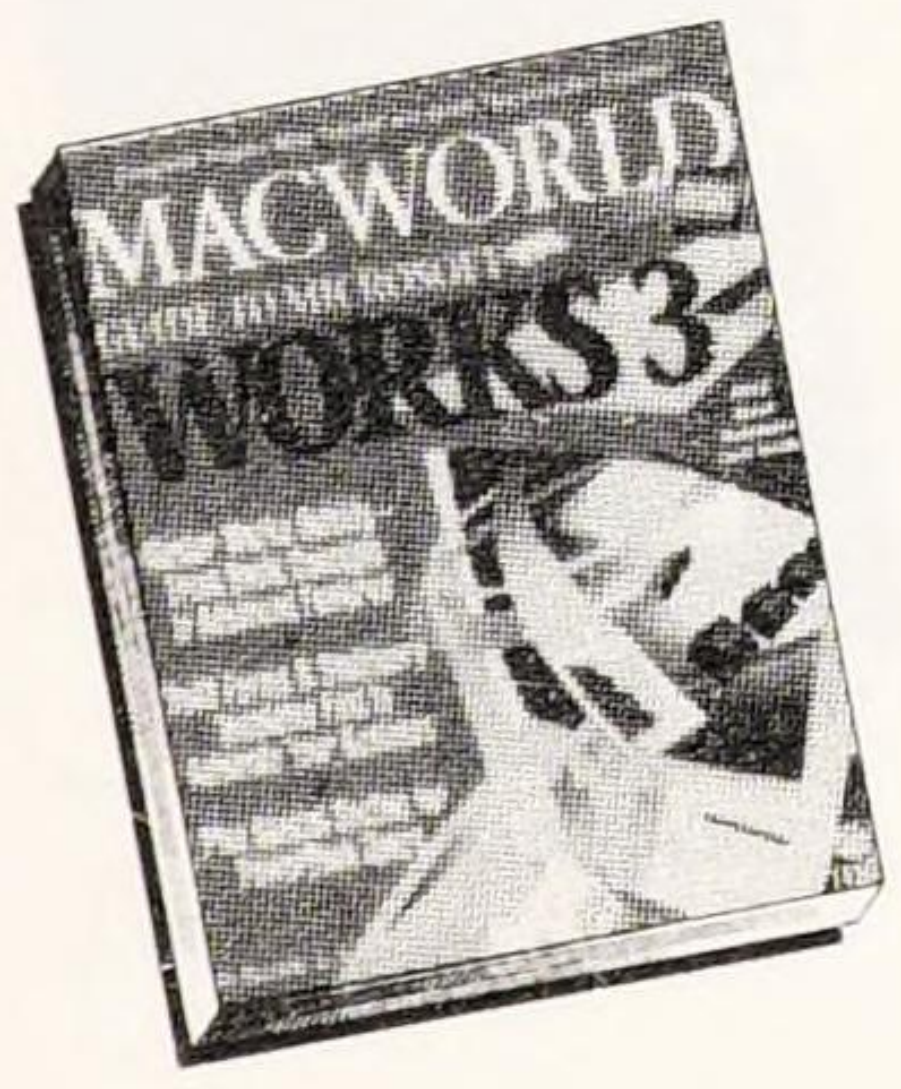


### Macworld Guide To Microsoft Word 5

*by Jim Heid, Macworld magazine's "Getting Started" columnist*

Learn Word the easy way with this *Macworld* Authorized Edition. Now updated for Word 5.1.

**$22.95 USA/$29.95 Canada/£20.95 UK & EIRE,**
**ISBN: 1-878058-39-8**

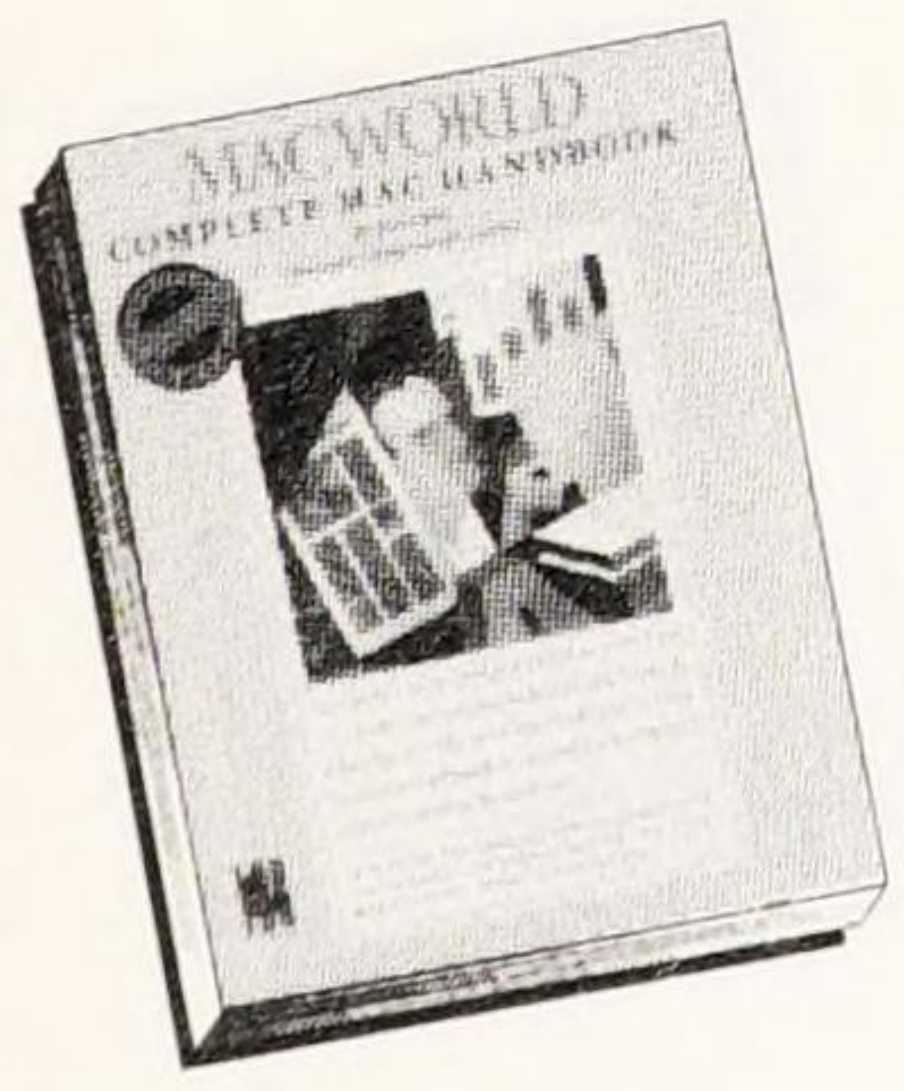


### Macworld Guide To Microsoft Works 3

*by Barrie A. Sosinsky*

Get inside the new Works so you can work more productively—the perfect blend of reference and tutorial.

**$22.95 USA/$29.95 Canada/£20.95 UK & EIRE,**
**ISBN: 1-878058-42-8**

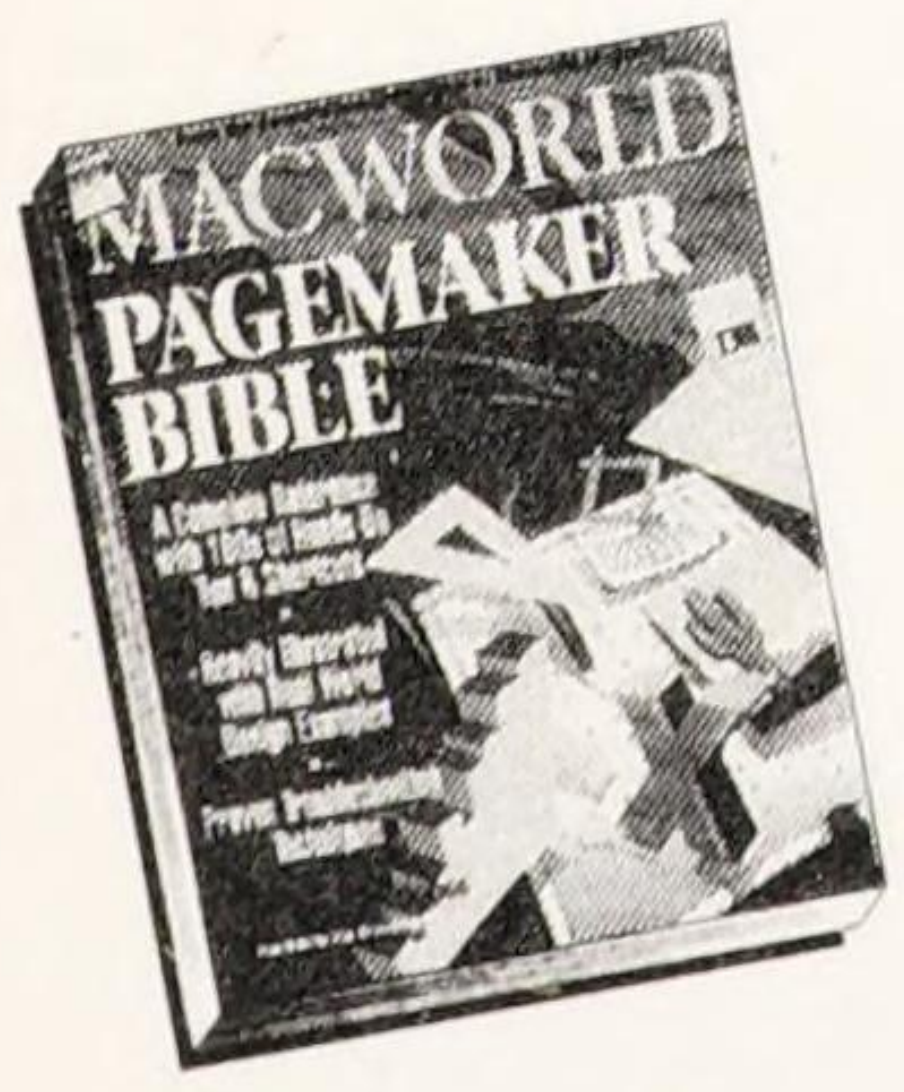


### Macworld Complete Mac Handbook

*by Jim Heid*

The most complete guide to getting started, mastering, and expanding your Mac.

**$26.95 USA/$35.95 Canada/£24.95 UK & EIRE,**
**ISBN: 1-878058-17-7**



### Macworld PageMaker Bible

*by Jo Ann Villalobos*

The ultimate insiders' guide to PageMaker 5, combining an authoritative and easy-to-use reference with tips and techniques. Includes 3 1/2" disk of templates.

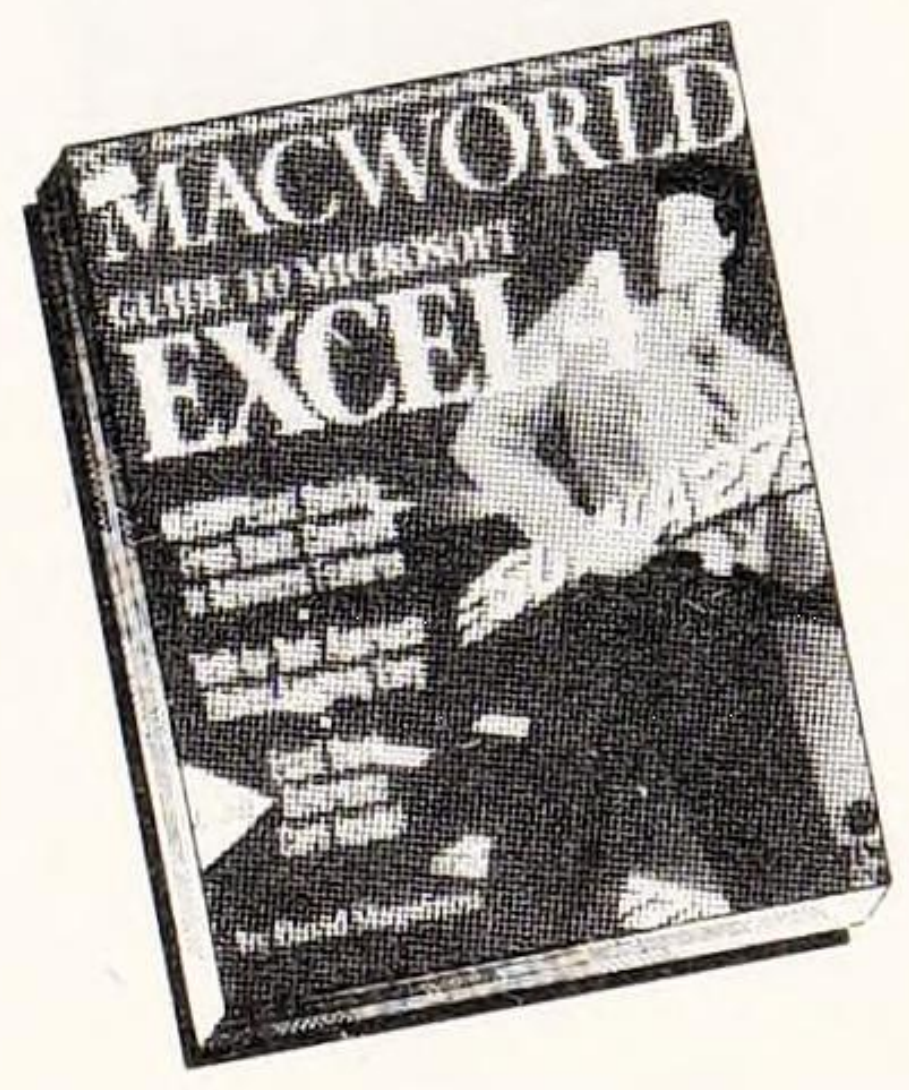
**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-84-3 — Available July 1993**

### Macworld Networking Handbook

*by David Kosiur, Ph.D.*

The ultimate insider's guide to Mac network management.

**$29.95 USA/$39.95 Canada/£27.45 UK & EIRE,**
**ISBN: 1-878058-31-2**

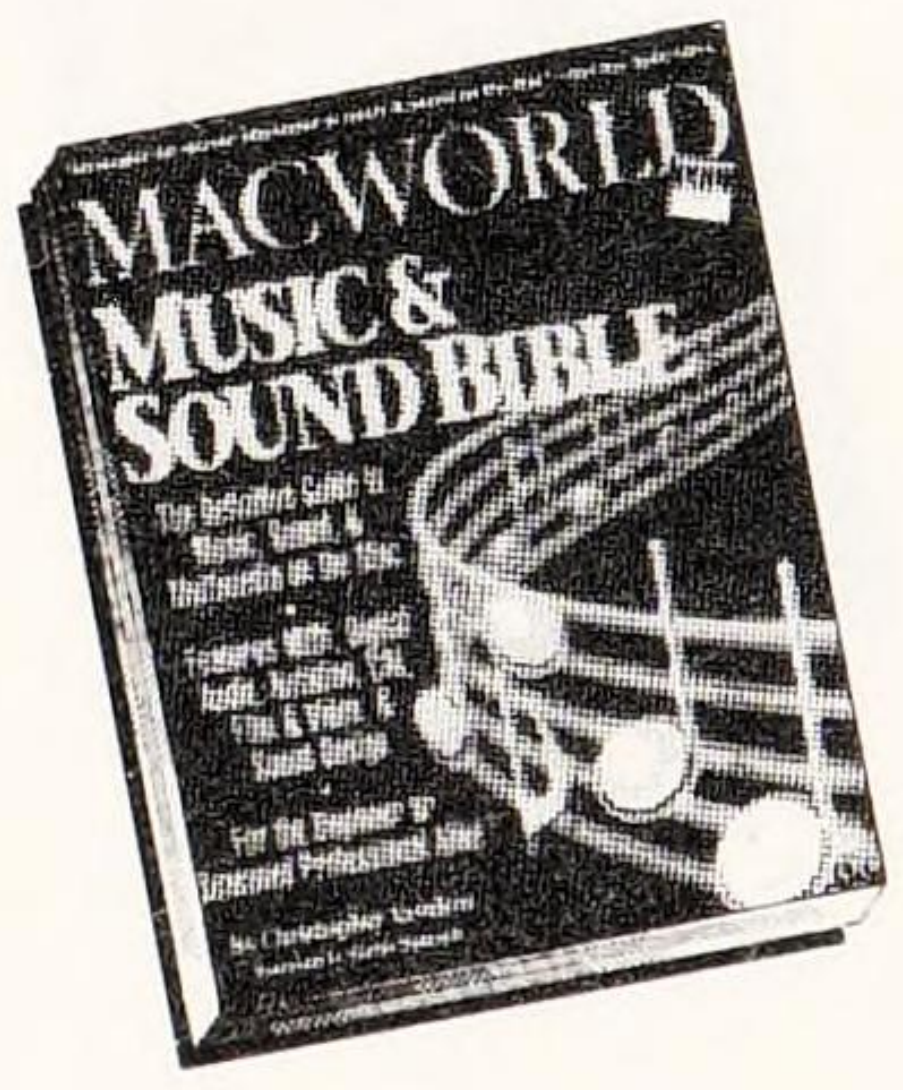


### Macworld Guide To Microsoft Excel 4

*by David Maguiness*

Build powerful spreadsheets quickly with this *Macworld* Authorized Edition to Excel 4.

**$22.95 USA/$29.95 Canada/£20.95 UK & EIRE,**
**ISBN: 1-878058-40-1**

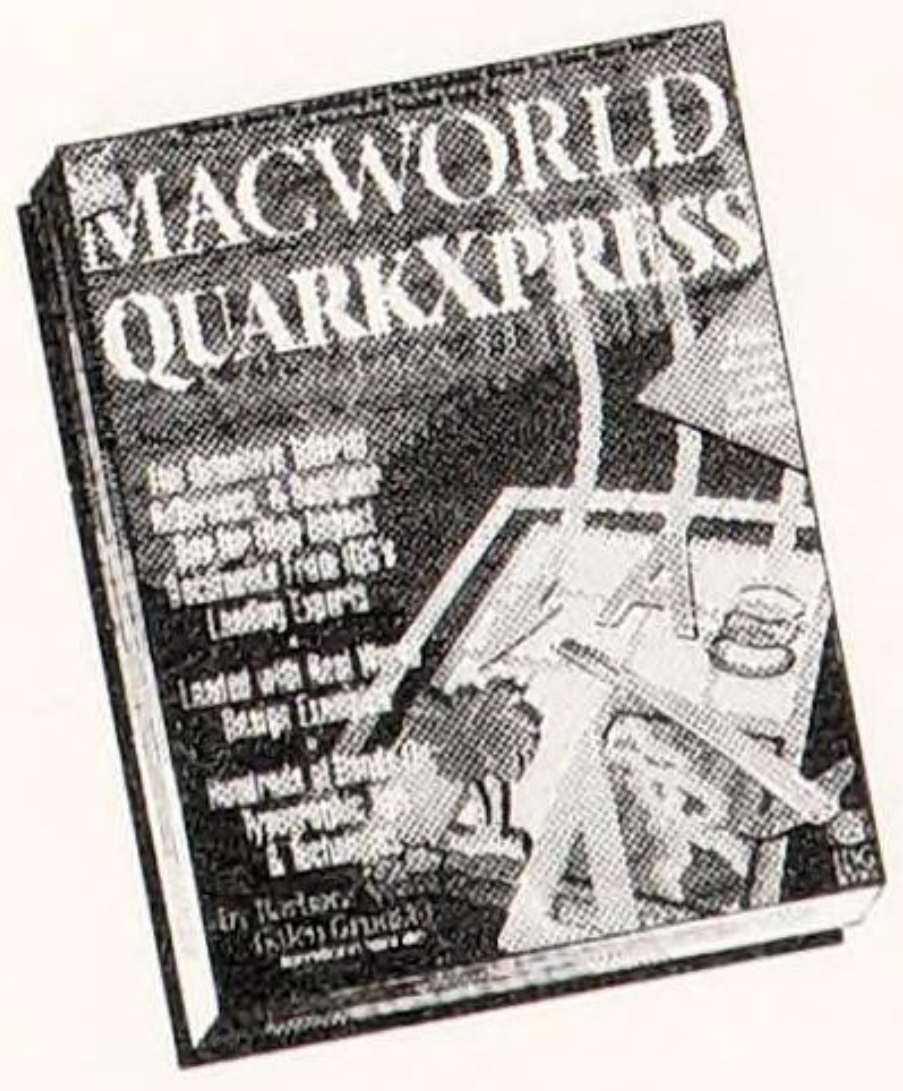


### Macworld Music & Sound Bible

*by Christopher Yavelow*

Finally, the definitive guide to music, sound, and multimedia on the Mac.

**$37.95 USA/$47.95 Canada/£34.95 UK & EIRE,**
**ISBN: 1-878058-18-5**



### Macworld QuarkXPress Designer Handbook

*by Barbara Assadi and Galen Gruman*

*Macworld* magazine's DTP experts help you master advanced features fast with this definitive tutorial, reference and designer tips resource on QuarkXPress.

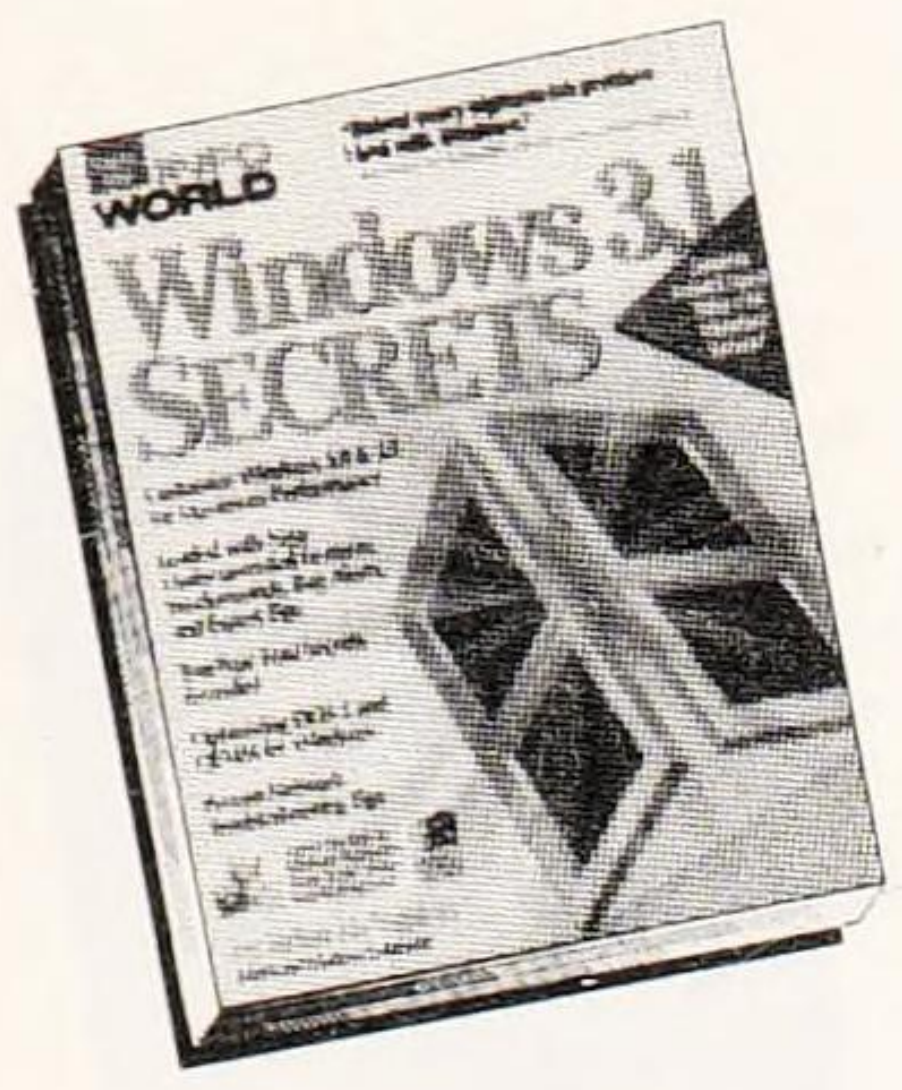
**$29.95 USA/$39.95 Canada/£27.45 UK & EIRE,**
**ISBN: 1-878058-85-1 — Available July 1993**

*For More Information Call 1-800-762-2974*

INFO WORLD TECHNICAL BOOKS

# InfoWorld SECRETS™ and Power Programming SECRETS™

SECRETS™ Books/disks offer a one-on-one session with an expert who reveals insider tips and techniques, undocumented features, and configuration and testing recommendations. The software includes ready-to-use utilities and applications that are strategically linked to the books' content. Power Programming SECRETS™ books/disks provide programming professionals with expert tips and techniques for successful and efficient application development. The included disk contains valuable templates and examples that users can easily customize to build their own applications.



## Windows 3.1 SECRETS™

*by Brian Livingston*

Windows expert Brian Livingston reveals the secret power of Windows 3.0 and 3.1— and The Best in Windows Shareware.

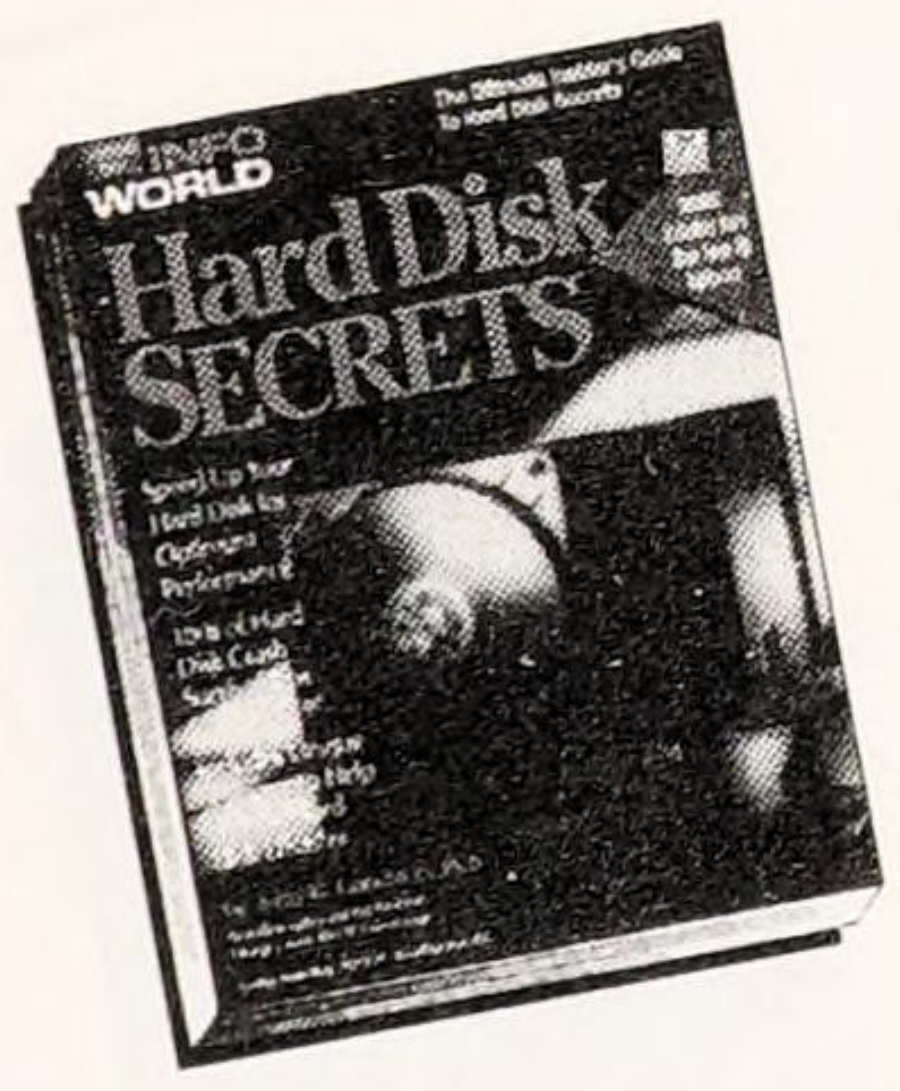
**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-43-6**

## Windows GIZMOS™

*by Brian Livingston and Margie Livingston*

The best Windows utilities, applications, and games — over 30 programs on 4 disks!

**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-66-5**



## Hard Disk SECRETS™

*by John M. Goodman, Ph.D., national bestselling author*

The insider's guide to maximizing hard disk performance. Includes disk of FREE hard disk tune-up software.

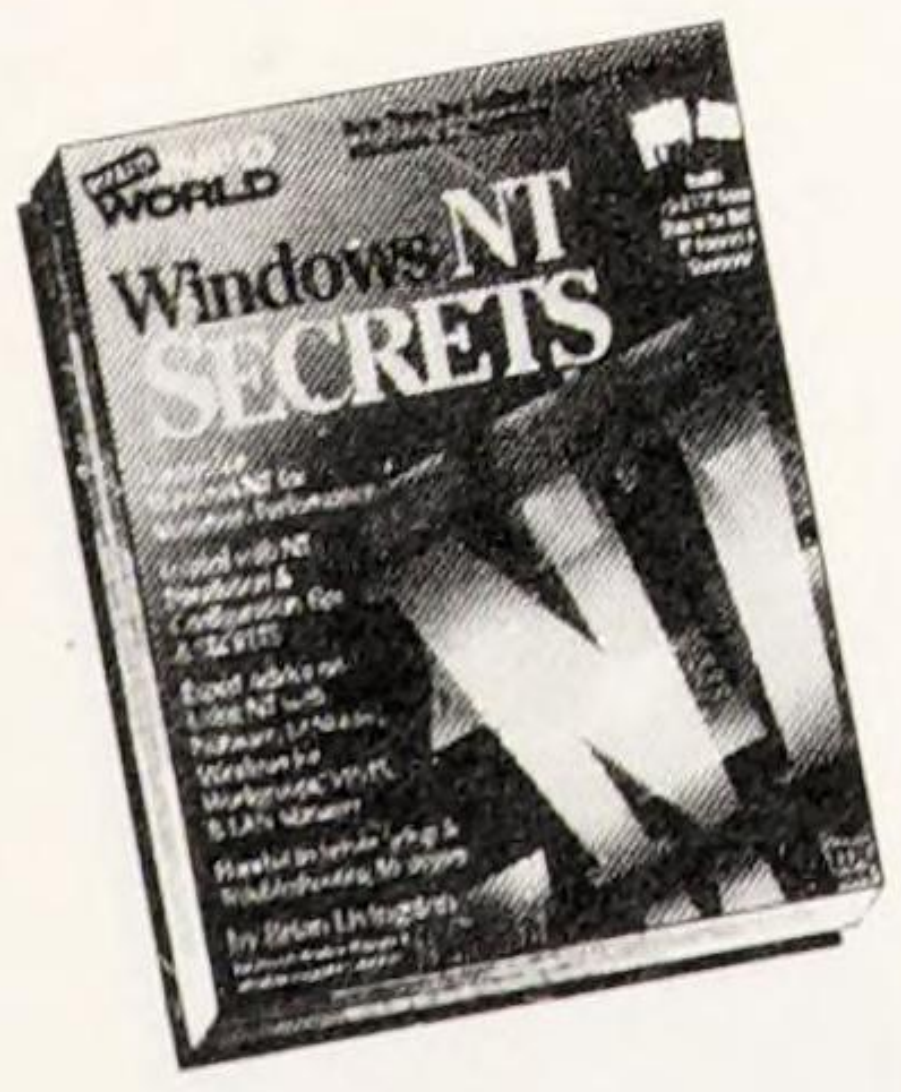
**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-64-9**

## Paradox 4 Power Programming SECRETS™, 2nd Edition

*by Gregory B. Salcedo and Martin W. Rudy*

Learn advanced Paradox 4 programming techniques with this completely revised and updated bestseller. Includes disk of valuable Paradox add-in tools.

**$44.95 USA/$59.95 Canada/£42.95 UK & EIRE,**
**ISBN: 1-878058-54-1**



## Windows NT SECRETS™

*by Brian Livingston*

A sequel to Livingston's bestselling book *Windows 3.1 SECRETS*. Focused on Microsoft's hot new multiuser/multitasking operating environment, this is the only book power users need to understand and master Windows NT.

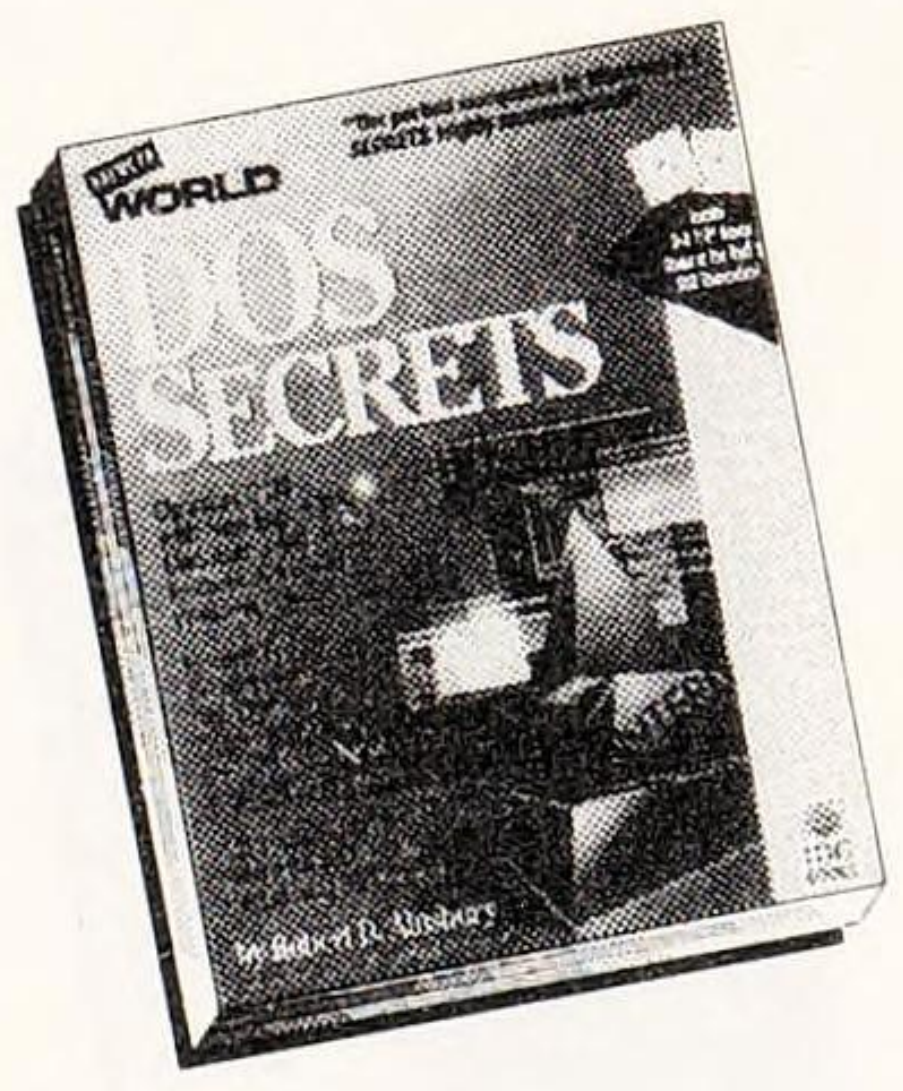
**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-71-1 — Available Sept. 1993**

## PC SECRETS™

*by Caroline M. Halliday*

IDG's technical support expert shows you how to optimize your PC's performance — includes two disks of valuable utilities.

**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-87805849-5**

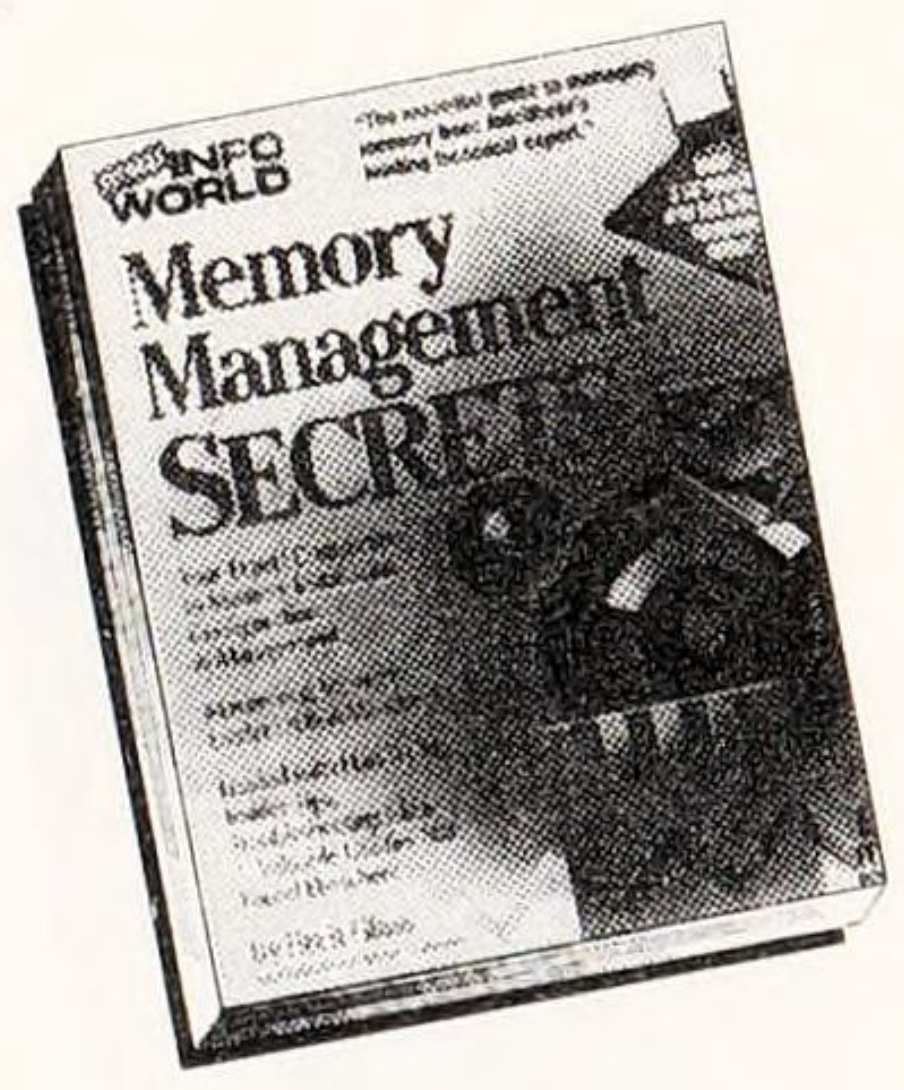


## DOS 6 SECRETS™

*by Robert D. Ainsbury*

Reveals hundreds of insider SECRETS and expert techniques to put the full power of DOS at your fingertips. The definitive reference for optimizing your PC with new DOS features. Includes coverage of the latest DOS extensions and 2 disks of the best in DOS Shareware.

**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-70-3**

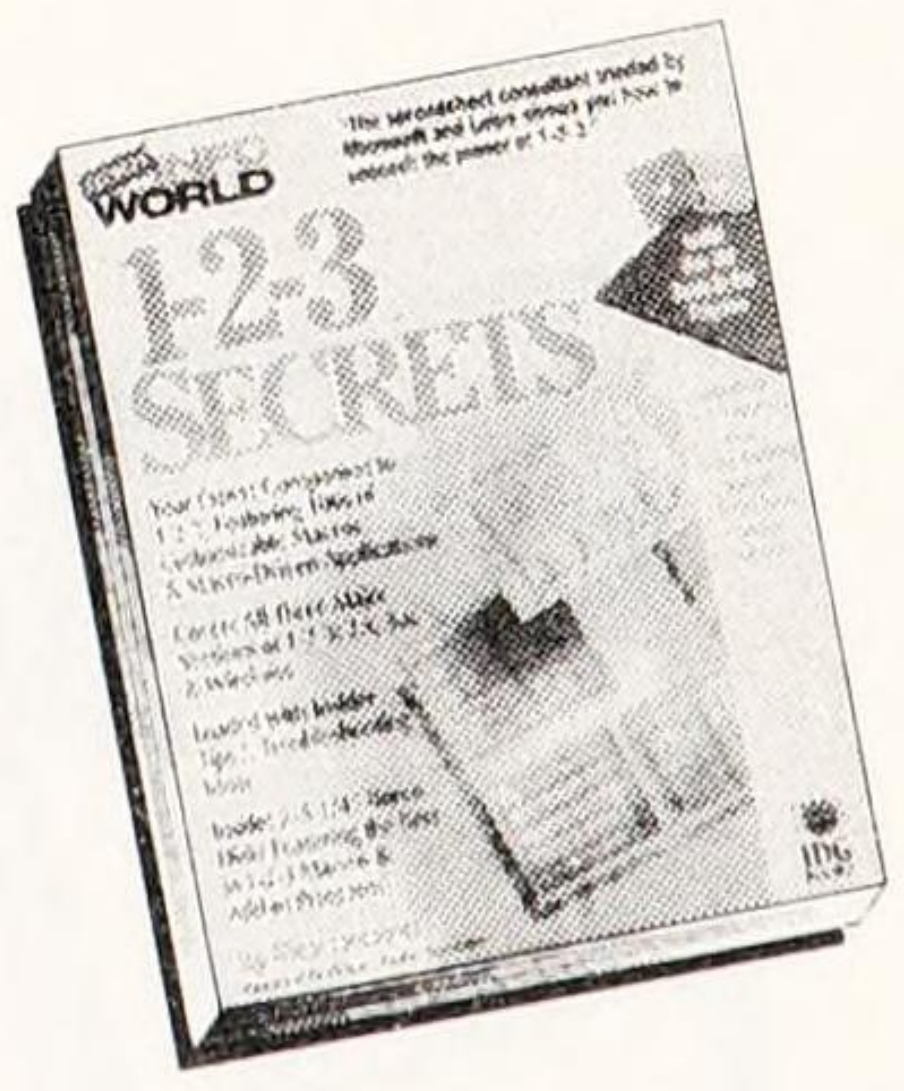


## Memory Management SECRETS™

*by Brett Glass*

InfoWorld's technical wizard Brett Glass reveals the hard-to-find and unusual secrets in this one-of-a-kind book/disk. Designed to help all users — beginning to advanced — install, configure, use, fine-tune, and understand memory management.

**$34.95 USA/$44.95 Canada/£32.95 UK & EIRE,**
**ISBN: 1-878058-72-X**



## 1-2-3 SECRETS™

*by Rick Dobson*

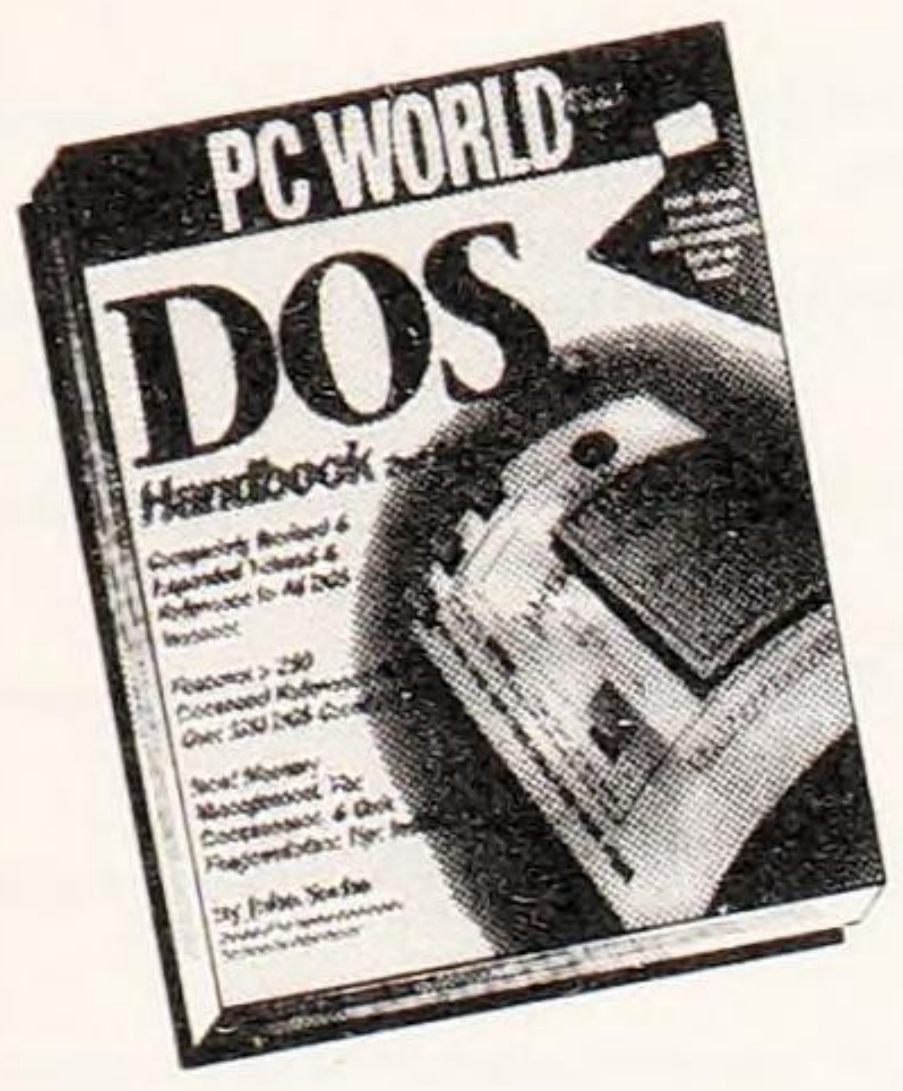
Let 1-2-3 expert Rick Dobson show you hundreds of undocumented tips, techniques and SECRETS to turbocharge 1-2-3 from simple spreadsheets to customized macro-driven applications.

**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-73-8**

*For More Information Call 1-800-762-2974*

# PC World Handbook

Expert information at your fingertips. Perfect for readers who need a complete tutorial of features as well as a reference to software applications and operating systems. All PC World Handbooks include bonus disks with software featuring useful templates, examples, and utilities that provide real value to the reader.

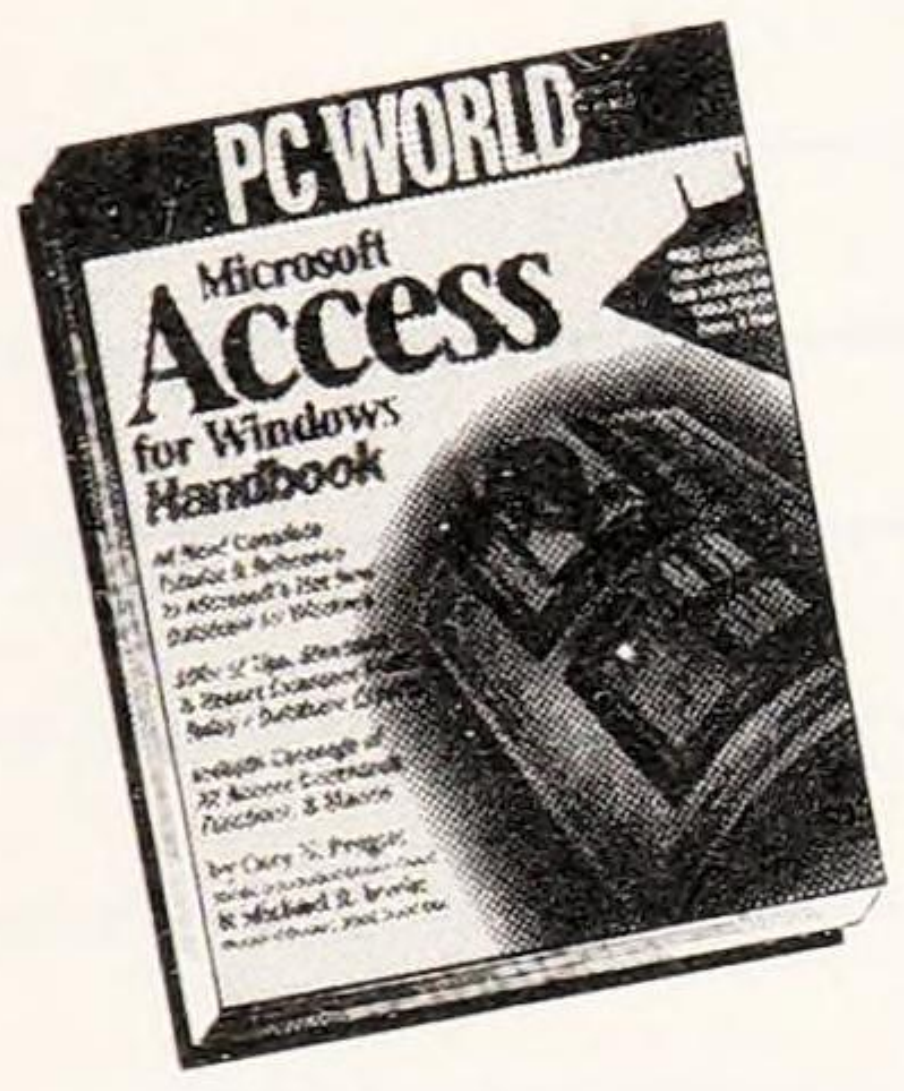


### PC World DOS 6 Handbook, 2nd Edition

*by John Socha, Clint Hicks, and Devra Hall*

Completely revised and updated! Includes extended features of DOS and the 250 page command reference that Microsoft excludes. A complete tutorial and reference PLUS Special Edition of Norton Commander software.

**$34.95 USA/$44.95 Canada/£32.95 UK & EIRE,**
**ISBN: 1-878058-79-7**



### PC World Microsoft Access Bible

*by Cary Prague and Michael Irwin*

This authoritative tutorial and reference on Microsoft's new Windows database is the perfect companion for every Microsoft Access user.

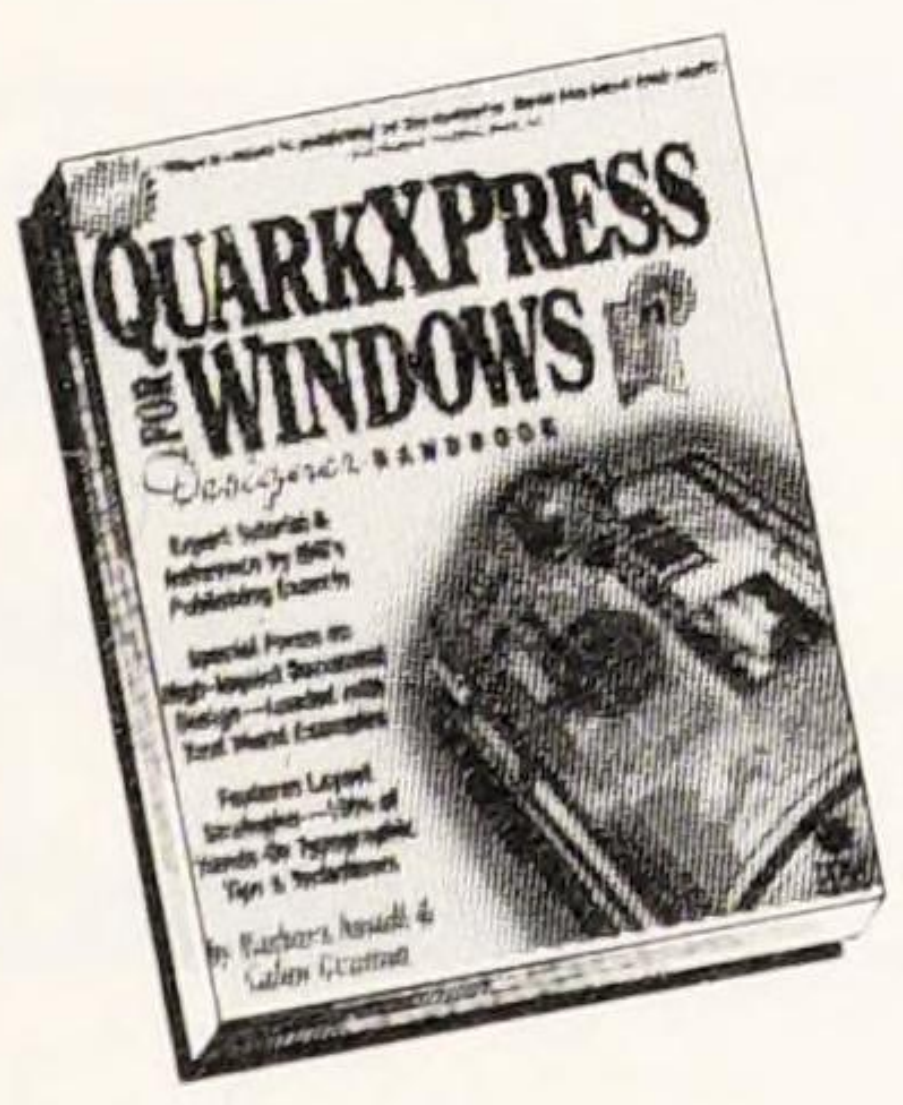
**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-81-9**

### Official XTree MS-DOS, Windows, and Hard Disk Management Companion, 3rd Edition

*by Beth Slick*

The only authorized guide to all versions of XTree, the most popular PC hard disk utility.

**$19.95 USA/$26.95 Canada/£18.45 UK & EIRE,**
**ISBN: 1-878058-57-6**

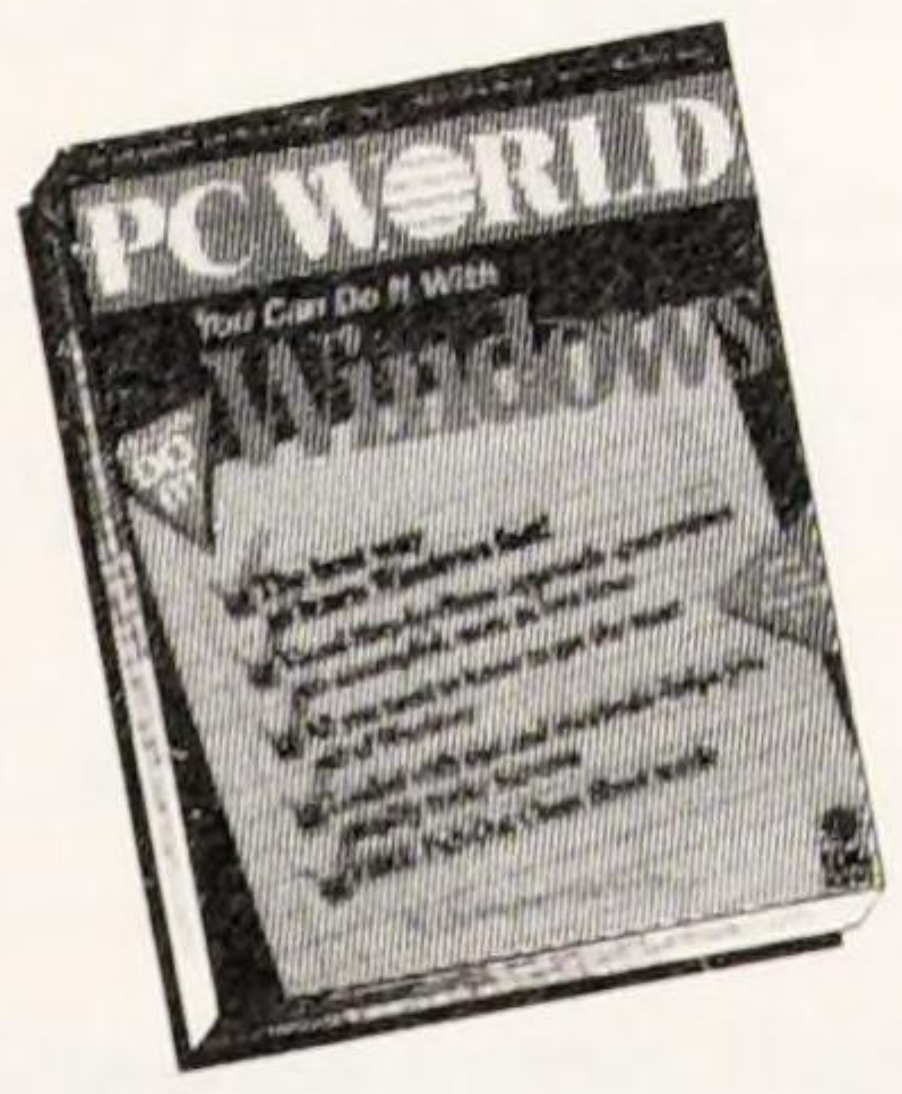


### QuarkXPress for Windows Designer Handbook

*by Barbara Assadi and Galen Gruman*

Make the move to QuarkXPress for Windows, the new professional desktop publishing powerhouse, with this expert reference and tutorial.

**$29.95 USA/$39.95 Canada/£27.45 UK & EIRE,**
**ISBN: 1-878058-45-2**

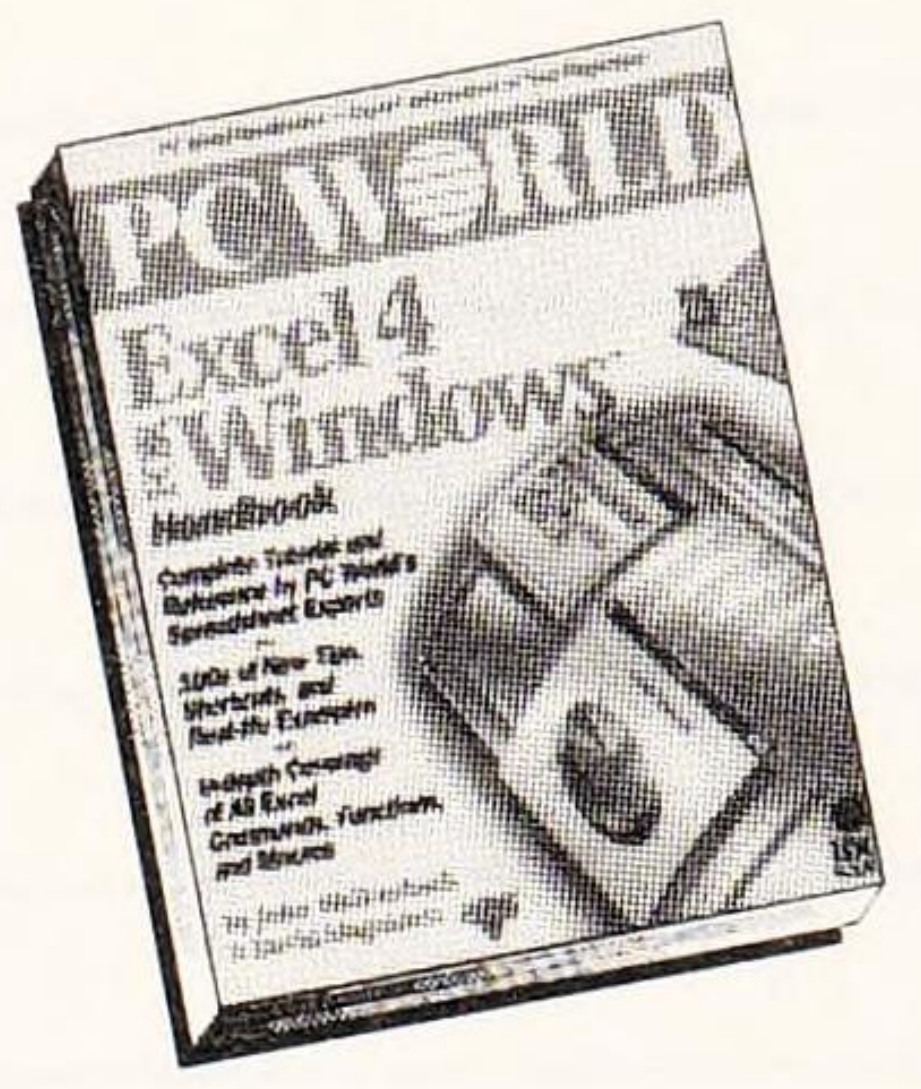


### PC World You Can Do It With Windows

*by Christopher Van Buren*

The best way to learn Window 3.1!

**$19.95 USA/$26.95 Canada/£18.45 VAT UK EIRE,**
**ISBN: 1-878058-37-1**

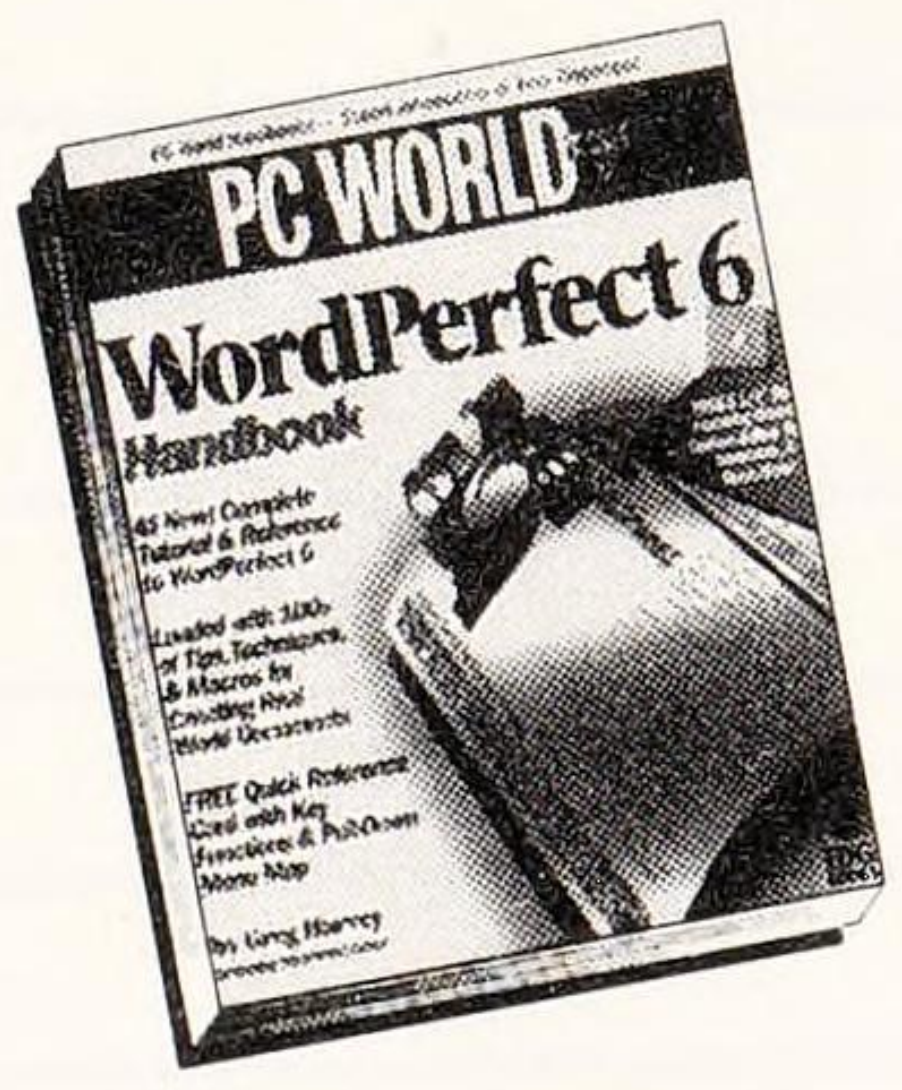


### PC World Excel 4 for Windows Handbook

*by John Walkenbach and David Maguiness*

Complete tutorial and reference by PC World's spreadsheet experts, with a FREE 32-page Function Reference booklet.

**$29.95 USA/$39.95 Canada/£27.45 UK & EIRE,**
**ISBN: 1-878058-46-0**

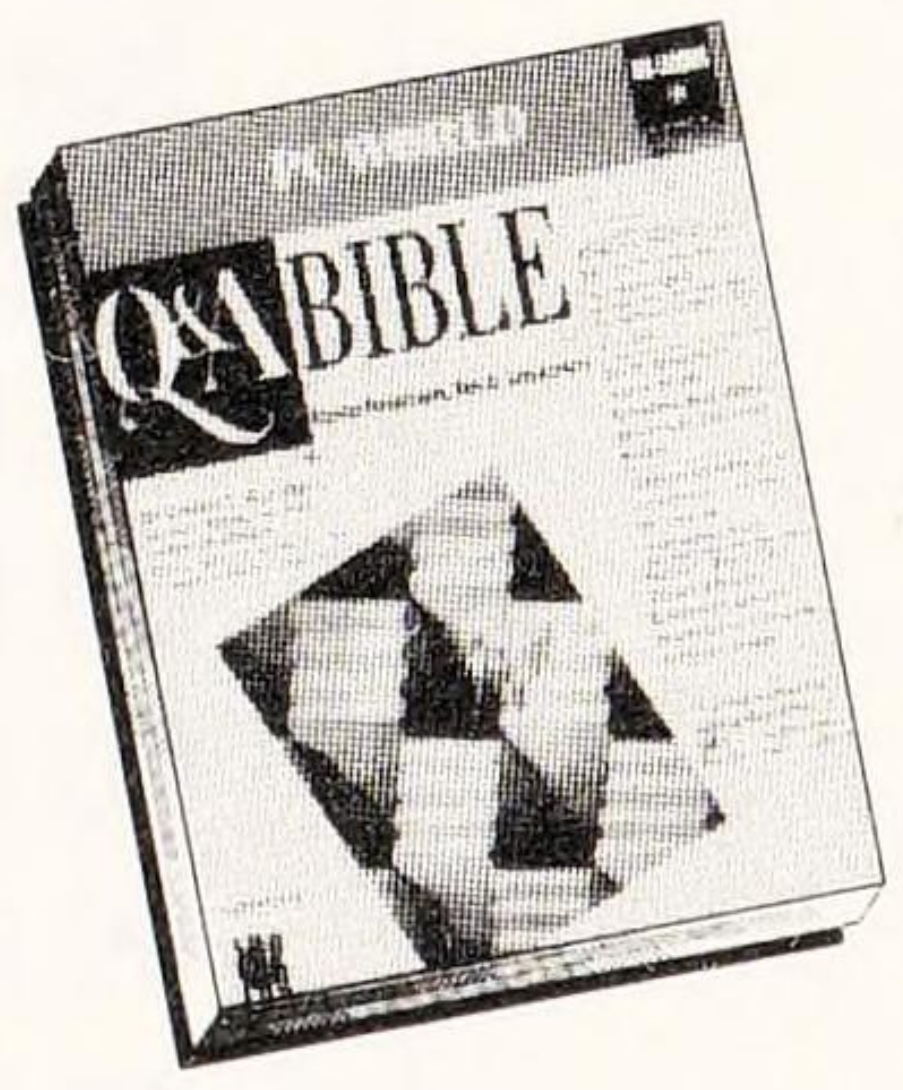


### PC World WordPerfect 6 Handbook

*by Greg Harvey*

Bestselling author and WordPerfect guru Greg Harvey brings you the ultimate tutorial and reference – complete with valuable software containing document templates, macros, and other handy WordPerfect tools.

**$34.95 USA/$44.95 Canada/£32.95 UK & EIRE,**
**ISBN: 1-878058-80-0 — Available July 1993**

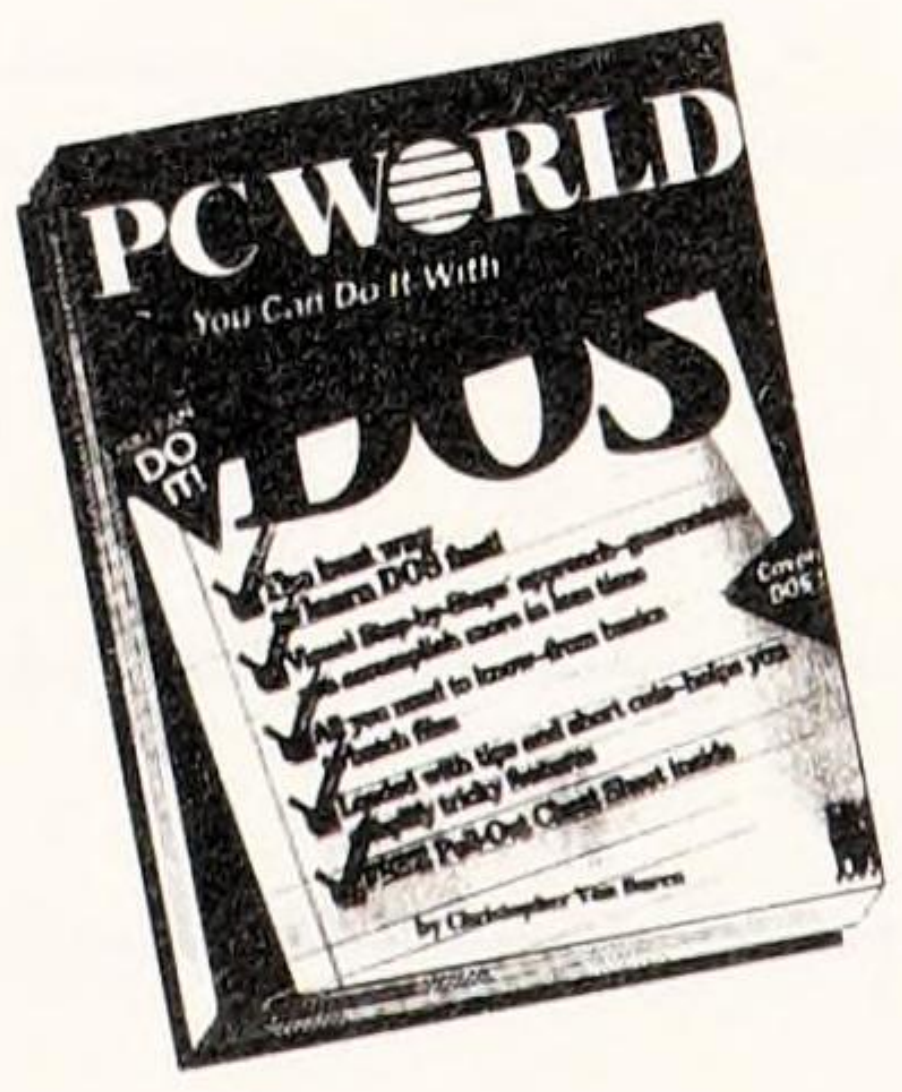


### PC World Q&A Bible, Version 4

*by Thomas J. Marcellus, Technical Editor of The Quick Answer*

The only thorough guide with a disk of databases for mastering Q&A Version 4.

**$39.95 USA/$52.95 Canada/£37.60 UK & EIRE,**
**ISBN: 1-878058-03-7**



### PC World You Can Do It With DOS

*by Christopher Van Buren*

The best way to learn DOS quickly and easily.

**$19.95 USA/$26.95 Canada/£18.45 VAT UK EIRE,**
**ISBN: 1-878058-38-X**

*For More Information Call 1-800-762-2974*

IDG
BOOKS

# Order Form

**Order Center: (800) 762-2974** (8 a.m.-5 p.m., PST, weekdays) or (415) 312-0600

**For Fastest Service:** Photocopy This Order Form and FAX it to : (415) 358-1260

| Quantity | ISBN | Title | Price | Total |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

**Shipping & Handling Charges**

| Subtotal | U.S. | Canada & International | International Air Mail |
|---|---|---|---|
| Up to $20.00 | Add $3.00 | Add $4.00 | Add $10.00 |
| $20.01-40.00 | $4.00 | $5.00 | $20.00 |
| $40.01-60.00 | $5.00 | $6.00 | $25.00 |
| $60.01-80.00 | $6.00 | $8.00 | $35.00 |
| Over $80.00 | $7.00 | $10.00 | $50.00 |

In U.S. and Canada, shipping is UPS ground or equivalent.
For Rush shipping call (800) 762-2974.

**Subtotal** __________

CA residents add applicable sales tax __________

IN residents add 5% sales tax __________

Canadian residents add 7% GST tax __________

Shipping __________

**TOTAL** __________

**Ship to:**

Name ____________________

Company ____________________

Address ____________________

City/State/Zip ____________________

Daytime Phone ____________________

Payment: ☐ Check to IDG Books (US Funds Only) ☐ Visa ☐ MasterCard ☐ American Express

Card # __________ Exp.______ Signature __________

Please send this order form to: IDG Books, 155 Bovet Road, San Mateo, CA 94402.
Allow up to 3 weeks for delivery. Thank you!

BOBS93

# Disclaimer and Copyright Notice

## Note

IDG Books Worldwide, Inc., warrants that the disks that accompany this book are free from defects in materials and workmanship for a period of 60 days from the date of purchase of this book. If IDG Books receives notification within the warranty period of defects in material or workmanship, IDG Books will replace the defective disks. The remedy for the breach of this warranty will be limited to replacement and will not encompass any other damages, including but not limited to loss of profit, and special, incidental, consequential, or other claims.

**5 ¼" Format Available.** The enclosed disks are in 3 ½-inch, 1.44MB format. If you don't have a drive that size or format, and cannot arrange to transfer the data to the disk size you need, you can obtain the programs on 5 ¼ 1.2MB disks by writing: IDG Books Worldwide, Attn: *DOS 6 SECRETS* Disks, IDG Books Worldwide, 155 Bovet Rd., Suite 310, San Mateo, CA 94402, or call 800-762-2974. Please allow 3–4 weeks for delivery.

**IDG Books Worldwide, InfoWorld Publishing Inc., and the author specifically disclaim all other warranties, express or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose with respect to defects in the disks, the programs, and source code contained therein, and/or the techniques described in the book, and in no event shall IDG Books Worldwide, InfoWorld, and/or the author be liable for any loss of profit or any other commercial damage, including but not limited to special, incidental, consequential, or other damages.**

## Licensing Agreement

Do not open the accompanying disk package until you have read and unless you agree with the terms of this licensing agreement. If you disagree and do not want to be bound by the terms of this licensing agreement, return the book for refund to the source from which you purchased it.

The contents of these disks are copyrighted and protected by both U.S. copyright law and international copyright treaty provisions. The individual programs on these disks are copyrighted by the authors of each program respectively. Each program has its own use permissions and limitations. You may copy any or all of these utilities to your computer system. To use each program, you must follow the individual requirements and restrictions detailed for each in the documentation contained in Part V of this book. Do not use a program if you do not want to follow its licensing agreement. Absolutely none of the material on these disks or listed in this book may ever be distributed, in original or modified form, for commercial purposes.

# Installation Bob's Better-Than-DOS Shareware Collection Disks

**Attention:** Before installing any of the programs from the distribution disks, read the Disclaimer and Copyright Notice on the previous page.

The programs included on the *DOS 6 SECRETS* disks are stored in compressed form. To transfer these programs to your hard drive, you *must* use the INSTALL program from Disk #1. You cannot install the software by using the DOS COPY command.

The INSTALL program allows you to specify the target drive and a parent directory (beneath which all the selected programs will be installed in individual subdirectories). Approximately 6MB of disk space is required to install all the programs. The space requirements for individual programs are listed in Part V.

The following steps detail the installation procedure:

**Step 1. Insert Disk #1.** Put the disk labeled *Disk 1* in either drive A or B, depending on the correct floppy drive size for your computer.

**Step 2. Run the INSTALL program.** From the DOS prompt, enter the command

```
a:install or b:install
```

**Step 3. Enter the target drive and directory.** Following a welcome screen, INSTALL prompts you to enter a drive and directory. Unless you specify otherwise, the drive C:\SECRETS is used by default.

**Step 4. Select the programs you want to try.** INSTALL will display a list of all the programs available on the disks. By default, every program is selected and marked with a check mark in the leftmost column. Use the up- and down-arrow keys to navigate through the list and use the spacebar to toggle the selection status of each individual program. After you have selected the desired programs, press the Enter key.

Each program is installed into a unique directory. Following is a list of all the directories created by the INSTALL program:

```
SECRETS ─┬─ ANARKEY
         ├─ ARJ
         ├─ BATUTILS
         ├─ CMFILER
         ├─ DUPE
         ├─ DOG
         ├─ EXTRADOS
         ├─ FORMASTR
         ├─ HDIRPLUS
         ├─ LCDFILES
         ├─ SECRETS
         ├─ SPINTOOL
         ├─ LIST
         ├─ FANSI
         ├─ PATRUTIL
         ├─ POCKETD
         ├─ QEDIT
         ├─ QFIND
         ├─ RELDEL
         ├─ SEEKEASY
         ├─ STACKEY
         ├─ UMBDRV
         └─ SCAN
```

**Step 5. Try out the software.** INSTALL does not make any modifications to your AUTOEXEC.BAT file, so none of the new directories will be added to the search path. The easiest way to evaluate a program is to change directories to the specific program's directory and then execute the program. Refer to Part V for further details about each program.

**Step 6. Register the programs you use.** If you like a program and continue to use it, register your copy. A copy of a registration form for each program is included in Part V.

**Inside! Author Bob Ainsbury has selected over 25 powerful shareware utilities that will help you get the most out of DOS 6! Programs include security, disk and file managers, batch file extenders, screen & keyboard extenders and memory management.**

# Bob's "Better Than DOS" Shareware Collection

## Multi-Purpose Toolboxes

**ExtraDOS Toolbox**
Batch file compiler and utilities

**Patriquin's Hard Disk Utilities**
Advanced DOS file management

**PocketD v1.64**
Directory listing and file management

## File & Directory Management

**ARJ**
File compression and archiving utility

**CMFILER**
Full-screen file manager

**HOTDIR PLUS**
Colorful and flexible directory lister

**LCD**
Directory changing utility

**RELDEL**
Permanent (non-undeletable) file eraser

**QEDIT**
Full screen text editor

**QFIND**
File finder

**List**
The ultimate file lister

## Disk Copying & Formatting

**FORMAT MASTER**
Full-screen disk formatting utility

**DUPE**
Fast disk duplication utility

## Hardware Enhancers

**DOG**
Disk defragger

**UMB_DRV.SYS**
Advanced memory management

**FANSI-CONSOLE**
Replacement for ANSI.SYS

**Spintest & SpinTime**
Hard disk analysis tools (Interleave & seek times)

**Stackey**
Keyboard enhancer

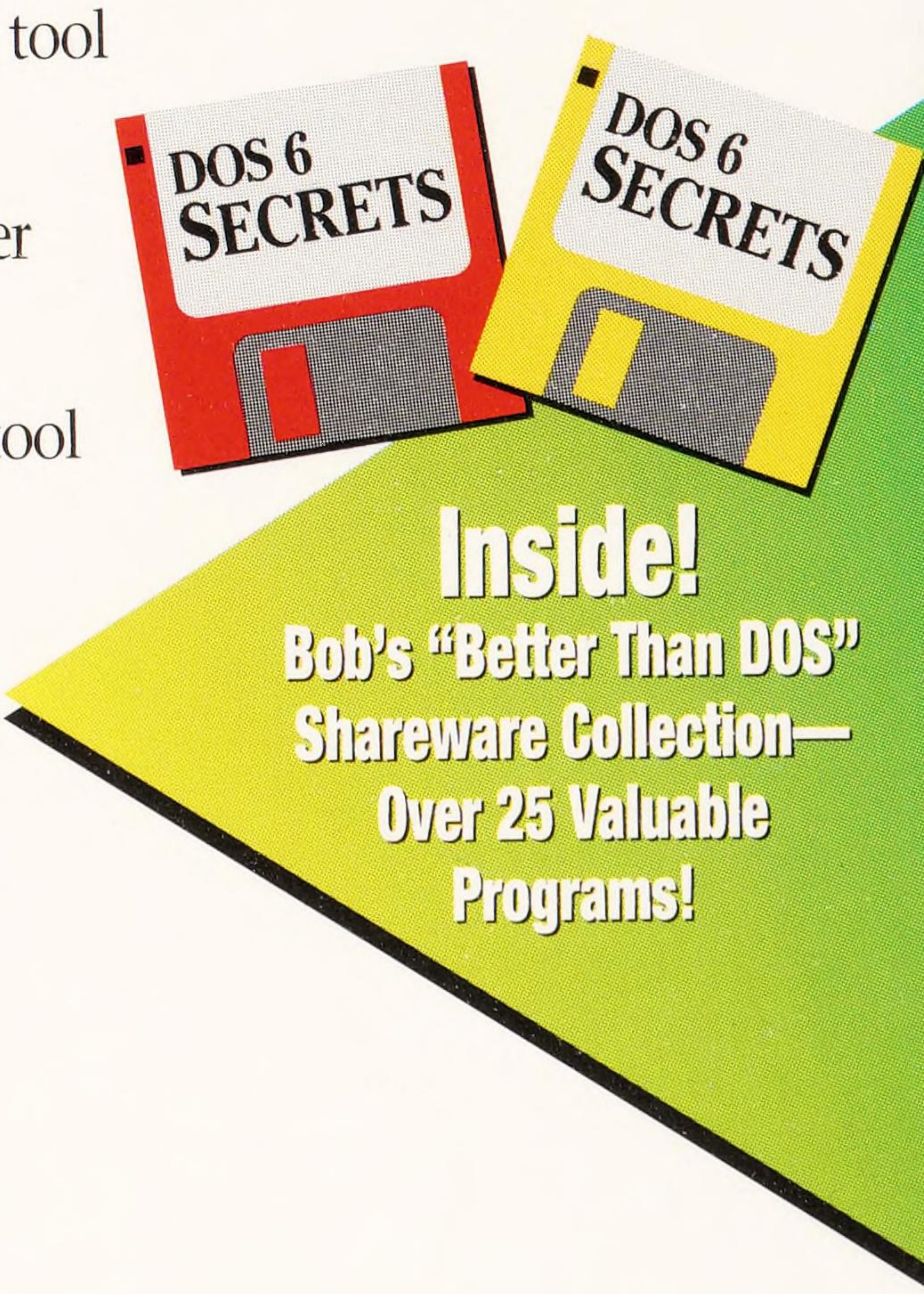
## PotPourri

**Anarkey**
A better DOSKEY command recall and editing tool

**Batutil**
Batch file extender

**SeekEasy**
A text searching tool

**Virus Utilities**
Anti-virus tools


DOS 6 SECRETS
DOS 6 SECRETS
Inside!
Bob's "Better Than DOS" Shareware Collection—
Over 25 Valuable Programs!

System Requirements: IBM PC Compatibles running DOS 3 or higher with at least 512K of memory

...SECRETS™ INFO WORLD

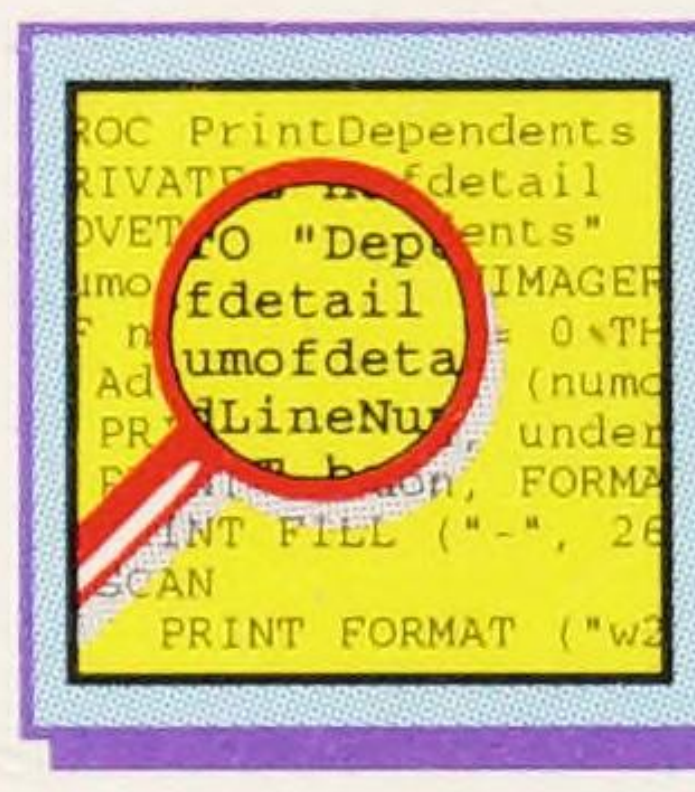
Ideal for DOS and Windows Users!

# DOS 6 SECRETS™

**"At last there is a book for long-time DOS users. An absolute must!"**
—Chris Smith, 1-2-3 for Windows Product Manager, Lotus Development Corp.

Unleash the power of DOS 6's new features and tools with this insider's companion to the best DOS ever. *DOS 6 SECRETS™* offers unique insights into the most powerful features of DOS 6, including DoubleSpace, DEFRAG, and memory optimization tools. In this one-on-one consulting session, DOS expert Bob Ainsbury reveals how to get the most out of DOS's new on-the-fly disk compression, defrag utilities, anti-virus tools, and more! All New! Covers DOS Releases 3-6.

## Inside *DOS 6 SECRETS* you'll get nearly 1000 pages of expert insights, tips, and SECRETS:



**Technical Review by**
**Linda Slovick,**
**InfoWorld Review Board**

- Find out which DOS 5 commands you shouldn't delete
- Learn the shortcomings of DoubleSpace floppies
- Learn why the QBASIC editor is better than plain EDIT
- Learn how to enhance the new CHOOSE command to accept Enter and Esc
- Check for any program in memory using a trick with MEM and FIND
- Squeeze out every last drop of available memory using the new optimization tools
- Understand, identify, and correct IRQ conflicts
- Learn how to run batch files once a day
- Learn how to customize and extend the DOS on-line help
- Transfer data with Interlink, DOS's new PC to laptop bridge

*DOS 6 SECRETS* also includes a complete and thorough command reference of every DOS 6 command and device driver, not available with Microsoft's DOS 6 documentation. If you're ready to maximize the power of DOS 6, this is the book for you!

Also available in the IDG Books' SECRETS Book/Disk Series: The bestselling *Windows 3.1 SECRETS, Hard Disk SECRETS, Memory Management SECRETS,* and *PC SECRETS.*

## About the Author

Bob Ainsbury, a software developer and computer book author, is vice president of software development at TechnoJock Software, Inc., which specializes in programming tools for DOS and Windows. He recently was the lead developer on the SmartPak product for 1-2-3 for Windows for Lotus Development Corporation.

**Another Bestselling ...SECRETS™ Book/Disk from IDG**

Level: Beginning to Advanced Users
Computer Book Shelving Category:
PCs/Operating Systems/DOS

***$39.95 US***
***$52.95 Canada***
***£37.60 incl VAT UK and Eire***

DOS 6 SECRETS
DOS 6 SECRETS

## Inside!

**Bob's "Better Than DOS" Shareware Collection— Over 25 Valuable Programs!**

***List Continued from Front Cover***

### Hardware Enhancers

**DOG**
**UMB__DRV.SYS**
**FANSI-CONSOLE**
**Spintest & SpinTime**
**Stackey**

### PotPourri

**Anarkey**
**Batutil**
**Seek Easy**
**Virus Utilities**

See inside back flap for more detailed descriptions of all programs.

IDG BOOKS

IDG Books Worldwide
An International Data Group Company
San Mateo, CA 94402

The World of Computer Knowledge

ISBN 1-878058-70-3

Printed in U.S.

9781878058706 22
11/01/2017 12:54-3